Stochastic Variance-Reduced Policy Gradient

Matteo Papini [*1] **Damiano Binaghi** [*1] **Giuseppe Canonaco** [*1] **Matteo Pirotta** [2] **Marcello Restelli** [1]

Abstract

In this paper, we propose a novel reinforcement-learning algorithm consisting in a stochastic variance-reduced version of policy gradient for solving Markov Decision Processes (MDPs). Stochastic variance-reduced gradient (SVRG) methods have proven to be very successful in supervised learning. However, their adaptation to policy gradient is not straightforward and needs to account for I) a non-concave objective function; II) approximations in the full gradient computation; and III) a non-stationary sampling process. The result is SVRPG, a stochastic variance-reduced policy gradient algorithm that leverages on importance weights to preserve the unbiasedness of the gradient estimate. Under standard assumptions on the MDP, we provide convergence guarantees for SVRPG with a convergence rate that is linear under increasing batch sizes. Finally, we suggest practical variants of SVRPG, and we empirically evaluate them on continuous MDPs.

1. Introduction

On a very general level, artificial intelligence addresses the problem of an agent that must select the right actions to solve a task. The approach of Reinforcement Learning (RL) (Sutton & Barto, 1998) is to learn the best actions by direct interaction with the environment and evaluation of the performance in the form of a reward signal. This makes RL fundamentally different from Supervised Learning (SL), where correct actions are explicitly prescribed by a human teacher (e.g., for classification, in the form of class labels). However, the two approaches share many challenges and tools. The problem of estimating a model from samples, which is at the core of SL, is equally fundamental in RL, whether we choose to model the environment, a value function, or directly a policy defining the agent's behaviour. Furthermore, when the tasks are characterized by large or continuous state-action spaces, RL needs the powerful function approximators (e.g., neural networks) that are the main subject of study of SL. In a typical SL setting, a performance function $J(\boldsymbol{\theta})$ has to be optimized w.r.t. to model parameters $\boldsymbol{\theta}$. The set of data that are available for training is often a subset of all the cases of interest, which may even be infinite, leading to optimization of finite sums that approximate the expected performance over an unknown data distribution. When generalization to the complete dataset is not taken into consideration, we talk about Empirical Risk Minimization (ERM). Even in this case, stochastic optimization is often used for reasons of efficiency. The idea of stochastic gradient (SG) ascent (Nesterov, 2013) is to iteratively focus on a random subset of the available data to obtain an approximate improvement direction. At the level of the single iteration, this can be much less expensive than taking into account all the data. However, the sub-sampling of data is a source of variance that can potentially compromise convergence, so that per-iteration efficiency and convergence rate must be traded off with proper handling of meta-parameters. Variance-reduced gradient algorithms such as SAG (Roux et al., 2012), SVRG (Johnson & Zhang, 2013) and SAGA (Defazio et al., 2014a) offer better ways of solving this trade-off, with significant results both in theory and practice. Although designed explicitly for ERM, these algorithms address a problem that affects more general machine learning problems.

In RL, stochastic optimization is rarely a matter of choice, since data must be actively sampled by interacting with an initially unknown environment. In this scenario, limiting the variance of the estimates is a necessity that cannot be avoided, which makes variance-reduced algorithms very interesting. Among RL approaches, policy gradient (Sutton et al., 2000) is the one that bears the closest similarity to SL solutions. The fundamental principle of these methods is to optimize a parametric policy through stochastic gradient ascent. Compared to other applications of SG, the cost of collecting samples can be very high since it requires to interact with the environment. This makes SVRG-like methods potentially much more efficient than, e.g., batch learning. Unfortunately, RL has a series of difficulties that are not present in ERM. First, in SL the objective can often be de-

[*]Equal contribution [1]Politecnico di Milano, Milano, Italy [2]Inria, Lille, France. Correspondence to: Matteo Papini <matteo.papini@polimi.it>.

Proceedings of the 35^{th} International Conference on Machine Learning, Stockholm, Sweden, PMLR 80, 2018.

signed to be strongly concave (we aim to maximize). This is not the case for RL, so we have to deal with non-concave objective functions. Then, as mentioned before, the dataset is not initially available and may even be infinite, which makes approximations unavoidable. This rules out SAG and SAGA because of their storage requirements, which leaves SVRG as the most promising choice. Finally, the distribution used to sample data is not under direct control of the algorithm designer, but it is a function of policy parameters that change over time as the policy is optimized, which is a form of non-stationarity. SVRG has been used in RL as an efficient technique for optimizing the per-iteration problem in Trust-Region Policy Optimization (Xu et al., 2017) or for policy evaluation (Du et al., 2017). In both the cases, the optimization problems faced resemble the SL scenario and are not affected by all the previously mentioned issues.

After providing background on policy gradient and SVRG in Section 2, we propose SVRPG, a variant of SVRG for the policy gradient framework, addressing all the difficulties mentioned above (see Section 3). In Section 4 we provide convergence guarantees for our algorithm, and we show a convergence rate that has an $O(1/T)$ dependence on the number T of iterations. In Section 5.2 we suggest how to set the meta-parameters of SVRPG, while in Section 5.3 we discuss some practical variants of the algorithm. Finally, in Section 7 we empirically evaluate the performance of our method on popular continuous RL tasks.

2. Preliminaries

In this section, we provide the essential background on policy gradient methods and stochastic variance-reduced gradient methods for finite-sum optimization.

2.1. Policy Gradient

A Reinforcement Learning task (Sutton & Barto, 1998) can be modelled with a discrete-time continuous Markov Decision Process (MDP) $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho\}$, where $\mathcal{S}$ is a continuous state space; $\mathcal{A}$ is a continuous action space; $\mathcal{P}$ is a Markovian transition model, where $\mathcal{P}(s'|s,a)$ defines the transition density from state s to s' under action a; $\mathcal{R}$ is the reward function, where $\mathcal{R}(s,a) \in [-R, R]$ is the expected reward for state-action pair (s,a); $\gamma \in [0,1)$ is the discount factor; and ρ is the initial state distribution. The agent's behaviour is modelled as a policy π, where $\pi(\cdot|s)$ is the density distribution over $\mathcal{A}$ in state s. We consider episodic MDPs with effective horizon H.[1] In this setting, we can limit our attention to trajectories of length H. A trajectory τ is a sequence of states and actions $(s_0, a_0, s_1, a_1, \dots, s_{H-1}, a_{H-1})$ observed by following a stationary policy, where $s_0 \sim \rho$. We denote with $p(\tau|\pi)$ the density distribution induced by policy π on the set $\mathcal{T}$ of all possible trajectories (see Appendix A for the definition), and with $\mathcal{R}(\tau)$ the total discounted reward provided by trajectory τ: $\mathcal{R}(\tau) = \sum_{t=0}^{H-1} \gamma^t \mathcal{R}(s_t, a_t)$. Policies can be ranked based on their expected total reward: $J(\pi) = \mathbb{E}_{\tau \sim p(\cdot|\pi)} [\mathcal{R}(\tau)|M]$. Solving an MDP M means finding $\pi^* \in \arg\max_\pi \{J(\pi)\}$.

Policy gradient methods restrict the search for the best performing policy over a class of parametrized policies $\Pi_{\boldsymbol{\theta}} = \{\pi_{\boldsymbol{\theta}} : \boldsymbol{\theta} \in \mathbb{R}^d\}$, with the only constraint that $\pi_{\boldsymbol{\theta}}$ is differentiable w.r.t. $\boldsymbol{\theta}$. For sake of brevity, we will denote the performance of a parametric policy with $J(\boldsymbol{\theta})$ and the probability of a trajectory τ with $p(\tau|\boldsymbol{\theta})$ (in some occasions, $p(\tau|\boldsymbol{\theta})$ will be replaced by $p_{\boldsymbol{\theta}}(\tau)$ for the sake of readability). The search for a locally optimal policy is performed through gradient ascent, where the policy gradient is (Sutton et al., 2000; Peters & Schaal, 2008a):

$$\nabla J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\tau \sim p(\cdot|\boldsymbol{\theta})} [\nabla \log p_{\boldsymbol{\theta}}(\tau) \mathcal{R}(\tau)] . \tag{1}$$

Notice that the distribution defining the gradient is induced by the current policy. This aspect introduces a nonstationarity in the sampling process. Since the underlying distribution changes over time, it is necessary to resample at each update or use weighting techniques such as importance sampling. Here, we consider the *online learning scenario*, where trajectories are sampled by interacting with the environment at each policy change. In this setting, stochastic gradient ascent is typically employed. At each iteration $k > 0$, a batch $\mathcal{D}_N^k = \{\tau_i\}_{i=0}^N$ of $N > 0$ trajectories is collected using policy $\pi_{\boldsymbol{\theta}_k}$. The policy is then updated as $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \widehat{\nabla}_N J(\boldsymbol{\theta}_k)$, where α is a step size and $\widehat{\nabla}_N J(\boldsymbol{\theta})$ is an estimate of Eq. (1) using $\mathcal{D}_N^k$. The most common policy gradient estimators (e.g., REINFORCE (Williams, 1992) and G(PO)MDP (Baxter & Bartlett, 2001)) can be expressed as follows

$$\widehat{\nabla}_N J(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} g(\tau_i|\boldsymbol{\theta}), \quad \tau_i \in \mathcal{D}_N^k, \tag{2}$$

where $g(\tau_i|\boldsymbol{\theta})$ is an estimate of $\nabla \log p_{\boldsymbol{\theta}}(\tau_i) \mathcal{R}(\tau_i)$. Although the REINFORCE definition is simpler than the G(PO)MDP one, the latter is usually preferred due to its lower variance. We refer the reader to Appendix A for details and a formal definition of g.

The main limitation of plain policy gradient is the high variance of these estimators. The naïve approach of increasing the batch size is not an option in RL due to the high cost of collecting samples, i.e., by interacting with the environment. For this reason, literature has focused on the introduction of baselines (i.e., functions $b : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$) aiming to reduce the variance (e.g., Williams, 1992; Peters & Schaal,

[1] The episode duration is a random variable, but the optimal policy can reach the target state (i.e., absorbing state) in less than H steps. This has not to be confused with a finite horizon problem where the optimal policy is non-stationary.

2008a; Thomas & Brunskill, 2017; Wu et al., 2018), see Appendix A for a formal definition of b. These baselines are usually designed to minimize the variance of the gradient estimate, but even them need to be estimated from data, partially reducing their effectiveness. On the other hand, there has been a surge of recent interest in variance reduction techniques for gradient optimization in supervised learning (SL). Although these techniques have been mainly derived for finite-sum problems, we will show in Section 3 how they can be used in RL. In particular, we will show that the proposed SVRPG algorithm can take the best of both worlds (i.e., SL and RL) since it can be plugged into a policy gradient estimate using baselines. The next section has the aim to describe variance reduction techniques for finite-sum problems. In particular, we will present the SVRG algorithm that is at the core of this work.

2.2. Stochastic Variance-Reduced Gradient

Finite-sum optimization is the problem of maximizing an objective function $f(\boldsymbol{\theta})$ which can be decomposed into the sum or average of a finite number of functions $z_i(\boldsymbol{\theta})$:

$$\max_{\boldsymbol{\theta}} \left\{ f(\boldsymbol{\theta}) = \frac{1}{N} \sum_{i=1}^{N} z_i(\boldsymbol{\theta}) \right\}.$$

This kind of optimization is very common in machine learning, where each z_i may correspond to a data sample x_i from a dataset $\mathcal{D}_N$ of size N (i.e., $z_i(\boldsymbol{\theta}) = z(x_i|\boldsymbol{\theta})$). A common requirement is that z must be smooth and concave in $\boldsymbol{\theta}$.[2] Under this hypothesis, full gradient (FG) ascent (Cauchy, 1847) with a constant step size achieves a linear convergence rate in the number T of iterations (i.e., parameter updates) (Nesterov, 2013). However, each iteration requires N gradient computations, which can be too expensive for large values of N. Stochastic Gradient (SG) ascent (e.g., Robbins & Monro, 1951; Bottou & LeCun, 2004) overcomes this problem by sampling a single sample x_i per iteration, but a vanishing step size is required to control the variance introduced by sampling. As a consequence, the lower per-iteration cost is paid with a worse, sub-linear convergence rate (Nemirovskii et al., 1983). Starting from SAG, a series of variations to SG have been proposed to achieve a better trade-off between convergence speed and cost per iteration: e.g., SAG (Roux et al., 2012), SVRG (Johnson & Zhang, 2013), SAGA (Defazio et al., 2014a), Finito (Defazio et al., 2014b), and MISO (Mairal, 2015). The common idea is to reuse past gradient computations to reduce the variance of the current estimate. In particular, Stochastic Variance-Reduced Gradient (SVRG) is often preferred to other similar methods for its limited storage requirements, which is a significant advantage when deep and/or wide neural networks are employed.

[2]Note that we are considering a maximization problem instead of the classical minimization one.

Algorithm 1 SVRG

Input: a dataset $\mathcal{D}_N$, number of epochs S, epoch size m, step size α, initial parameter $\boldsymbol{\theta}_m^0 := \widetilde{\boldsymbol{\theta}}^0$
for $s = 0$ **to** $S-1$ **do**
  $\boldsymbol{\theta}_0^{s+1} := \widetilde{\boldsymbol{\theta}}^s = \boldsymbol{\theta}_m^s$
  $\widetilde{\mu} = \nabla f(\widetilde{\boldsymbol{\theta}}^s)$
  for $t = 0$ **to** $m-1$ **do**
    $x \sim \mathcal{U}(\mathcal{D}_N)$
    $v_t^{s+1} = \widetilde{\mu} + \nabla z(x|\boldsymbol{\theta}_t^{s+1}) - \nabla z(x|\widetilde{\boldsymbol{\theta}}^s)$
    $\boldsymbol{\theta}_{t+1}^{s+1} = \boldsymbol{\theta}_t^{s+1} + \alpha v_t^{s+1}$
  end for
end for
Concave case: **return** $\boldsymbol{\theta}_m^S$
Non-Concave case: **return** $\boldsymbol{\theta}_t^{s+1}$ with (s,t) picked uniformly at random from $\{[0, S-1] \times [0, m-1]\}$

The idea of SVRG (Algorithm 1) is to alternate full and stochastic gradient updates. Each $m = O(N)$ iterations, a snapshot $\widetilde{\boldsymbol{\theta}}$ of the current parameter is saved together with its full gradient $\nabla f(\widetilde{\boldsymbol{\theta}}) = \frac{1}{N}\sum_i \nabla z(x_i|\widetilde{\boldsymbol{\theta}})$. Between snapshots, the parameter is updated with $\blacktriangledown f(\boldsymbol{\theta})$, a gradient estimate corrected using stochastic gradient. For any $t \in \{0, \dots, m-1\}$:

$$\blacktriangledown f(\boldsymbol{\theta}_t) := v_t = \nabla f(\widetilde{\boldsymbol{\theta}}) + \nabla z(x|\boldsymbol{\theta}_t) - \nabla z(x|\widetilde{\boldsymbol{\theta}}), \quad (3)$$

where x is sampled uniformly at random from $\mathcal{D}_N$ (i.e., $x \sim \mathcal{U}(\mathcal{D}_N)$). Note that $t = 0$ corresponds to a FG step (i.e., $\blacktriangledown f(\boldsymbol{\theta}_0) = \nabla f(\widetilde{\boldsymbol{\theta}})$) since $\boldsymbol{\theta}_0 := \widetilde{\boldsymbol{\theta}}$. The corrected gradient $\blacktriangledown f(\boldsymbol{\theta})$ is an unbiased estimate of $\nabla f(\boldsymbol{\theta})$, and it is able to control the variance introduced by sampling even with a fixed step size, achieving a linear convergence rate without resorting to a plain full gradient.

More recently, some extensions of variance reduction algorithms to the non-concave objectives have been proposed (e.g., Allen-Zhu & Hazan, 2016; Reddi et al., 2016a;b). In this scenario, f is typically required to be L-smooth, i.e., $\left\|\nabla f(\boldsymbol{\theta}') - \nabla f(\boldsymbol{\theta})\right\|_2 \leq L \left\|\boldsymbol{\theta}' - \boldsymbol{\theta}\right\|_2$ for each $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{R}^n$ and for some Lipschitz constant L. Under this hypothesis, the convergence rate of SG is $O(1/\sqrt{T})$ (Ghadimi & Lan, 2013), i.e., $T = O(1/\epsilon^2)$ iterations are required to get $\|\nabla f(\boldsymbol{\theta})\|_2^2 \leq \epsilon$. Again, SVRG achieves the same rate as FG (Reddi et al., 2016a), which is $O(\frac{1}{T})$ in this case (Nesterov, 2013). The only additional requirement is to select $\boldsymbol{\theta}^*$ uniformly at random among all the $\boldsymbol{\theta}_k$ instead of simply setting it to the final value (k being the iterations).

3. SVRG in Reinforcement Learning

In online RL problems, the usual approach is to tune the batch size of SG to find the optimal trade-off between variance and speed. Recall that, compared to SL, the samples

are not fixed in advance but we need to collect them at each policy change. Since this operation may be costly, we would like to minimize the number of interactions with the environment. For these reasons, we would like to apply SVRG to RL problems in order to limit the variance introduced by sampling trajectories, which would ultimately lead to faster convergence. However, a direct application of SVRG to RL is not possible due to the following issues:

Non-concavity: the objective function $J(\boldsymbol{\theta})$ is typically non-concave.

Infinite dataset: the RL optimization cannot be expressed as a finite-sum problem. The objective function is an expected value over the trajectory density $p_{\boldsymbol{\theta}}(\tau)$ of the total discounted reward, for which we would need an infinite dataset.

Non-stationarity: the distribution of the samples changes over time. In particular, the value of the policy parameter $\boldsymbol{\theta}$ influences the sampling process.

To deal with non-concavity, we require $J(\boldsymbol{\theta})$ to be L-smooth, which is a reasonable assumption for common policy classes such as Gaussian[3] and softmax (e.g., Furmston & Barber, 2012; Pirotta et al., 2015). Because of the infinite dataset, we can only rely on an estimate of the full gradient. Harikandeh et al. (2015) analysed this scenario under the assumptions of z being concave, showing that SVRG is robust to an inexact computation of the full gradient. In particular, it is still possible to recover the original convergence rate if the error decreases at an appropriate rate. Bietti & Mairal (2017) performed a similar analysis on MISO. In Section 4 we will show how the estimation accuracy impacts on the convergence results with a non-concave objective. Finally, the non-stationarity of the optimization problem introduces a bias into the SVRG estimator in Eq. (3). To overcome this limitation we employ importance weighting (e.g., Rubinstein, 1981; Precup, 2000) to correct the distribution shift.

We can now introduce Stochastic Variance-Reduced Policy Gradient (SVRPG) for a generic policy gradient estimator g. Pseudo-code is provided in Algorithm 2. The overall structure is the same as Algorithm 1, but the snapshot gradient is not exact and the gradient estimate used between snapshots is corrected using importance weighting:[4]

$$\blacktriangledown J(\boldsymbol{\theta}_t) = \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}) + g(\tau|\boldsymbol{\theta}_t) - \omega(\tau|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}}) g(\tau|\widetilde{\boldsymbol{\theta}})$$

for any $t \in \{0, \dots, m-1\}$, where $\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}})$ is as in Eq. (2) where $\mathcal{D}_N$ is sampled using the snapshot policy $\pi_{\widetilde{\boldsymbol{\theta}}}$, τ is sampled from the current policy $\pi_{\boldsymbol{\theta}_t}$, and $\omega(\tau|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}}) = \frac{p(\tau|\widetilde{\boldsymbol{\theta}})}{p(\tau|\boldsymbol{\theta}_t)}$ is an importance weight from $\pi_{\boldsymbol{\theta}_t}$ to the snapshot policy $\pi_{\widetilde{\boldsymbol{\theta}}}$. Similarly to SVRG, we have that $\boldsymbol{\theta}_0 := \widetilde{\boldsymbol{\theta}}$, and the update is a FG step. Our update is still fundamentally on-policy since the weighting concerns only the correction term. However, this partial "off-policyness" represents an additional source of variance. This is a well-known issue of importance sampling (e.g., Thomas et al., 2015). To mitigate it, we use mini-batches of trajectories of size $B \ll N$ to average the correction, i.e.,

$$\blacktriangledown J(\boldsymbol{\theta}_t) := v_t = \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}) + \underbrace{\frac{1}{B}\sum_{i=0}^{B-1}\left[g(\tau_i|\boldsymbol{\theta}_t) - \omega(\tau_i|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}}) g(\tau_i|\widetilde{\boldsymbol{\theta}})\right]}_{c_t}. \tag{4}$$

It is worth noting that the full gradient and the correction term have the same expected value: $\mathbb{E}_{\tau_i \sim p(\cdot|\boldsymbol{\theta}_t)}\left[\frac{1}{B}\sum_{i=0}^{B-1} \omega(\tau_i|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}}) g(\tau_i|\widetilde{\boldsymbol{\theta}})\right] = \nabla J(\widetilde{\boldsymbol{\theta}})$.[5] This property will be used to prove Lemma 3.1. The use of mini-batches is also common practice in SVRG since it can yield a performance improvement even in the supervised case (Harikandeh et al., 2015; Konečnỳ et al., 2016). It is easy to show that the SVRPG estimator has the following, desirable properties:

Lemma 3.1. *Let $\widehat{\nabla}_N J(\boldsymbol{\theta})$ be an unbiased estimator of (1) and let $\boldsymbol{\theta}^* \in \arg\min_{\boldsymbol{\theta}}\{J(\boldsymbol{\theta})\}$. Then, the SVRG estimate*

Algorithm 2 SVRPG

Input: number of epochs S, epoch size m, step size α, batch size N, mini-batch size B, gradient estimator g, initial parameter $\boldsymbol{\theta}_m^0 := \widetilde{\boldsymbol{\theta}}^0 := \boldsymbol{\theta}_0$
for $s = 0$ **to** $S-1$ **do**
  $\boldsymbol{\theta}_0^{s+1} := \widetilde{\boldsymbol{\theta}}^s = \boldsymbol{\theta}_m^s$
  Sample N trajectories $\{\tau_j\}$ from $p(\cdot|\widetilde{\boldsymbol{\theta}}^s)$
  $\widetilde{\mu} = \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)$ (see Eq. (2))
  for $t = 0$ **to** $m-1$ **do**
    Sample B trajectories $\{\tau_i\}$ from $p(\cdot|\boldsymbol{\theta}_t^{s+1})$
    $c_t^{s+1} = \frac{1}{B}\sum_{i=0}^{B-1}\Big(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i|\boldsymbol{\theta}_t^{s+1}, \widetilde{\boldsymbol{\theta}}^s) g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\Big)$
    $v_t^{s+1} = \widetilde{\mu} + c_t^{s+1}$
    $\boldsymbol{\theta}_{t+1}^{s+1} = \boldsymbol{\theta}_t^{s+1} + \alpha v_t^{s+1}$
  end for
end for
return $\boldsymbol{\theta}_A := \boldsymbol{\theta}_t^{s+1}$ with (s,t) picked uniformly at random from $\{[0, S-1] \times [0, m-1]\}$

[3] See Appendix C for more details on the Gaussian policy case.

[4] Note that g can be any unbiased estimator, with or without baseline. The unbiasedness is required for theoretical results (e.g., Appendix A).

[5] The reader can refer to Appendix A for off-policy gradients and variants of REINFORCE and G(PO)MDP.

in (4) *is* unbiased

$$\mathbb{E}\left[\blacktriangledown J(\boldsymbol{\theta})\right] = \nabla J(\boldsymbol{\theta}). \tag{5}$$

and regardless of the mini-batch size B*:*[6]

$$\mathbb{V}\mathrm{ar}\left[\blacktriangledown J(\boldsymbol{\theta}^*)\right] = \mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_N J(\boldsymbol{\theta}^*)\right]. \tag{6}$$

Previous results hold for both REINFORCE and G(PO)MDP. In particular, the latter result suggests that an SVRG-like algorithm using $\blacktriangledown J(\boldsymbol{\theta})$ can achieve faster convergence, by performing much more parameter updates with the same data without introducing additional variance (at least asymptotically). Note that the randomized return value of Algorithm 2 does not affect online learning at all, but will be used as a theoretical tool in the next section.

4. Convergence Guarantees of SVRPG

In this section, we state the convergence guarantees for SVRPG with REINFORCE or G(PO)MDP gradient estimator. We mainly leverage on the recent analysis of non-concave SVRG (Reddi et al., 2016a; Allen-Zhu & Hazan, 2016). Each of the three challenges presented at the beginning of Section 3 can potentially prevent convergence, so we need additional assumptions. In Appendix C we show how Gaussian policies satisfy these assumptions.

1) Non-concavity. A common assumption, in this case, is to assume the objective function to be L-smooth. However, in RL we can consider the following assumption which is sufficient for the L-smoothness of the objective (see Lemma B.2).

Assumption 4.1 (On policy derivatives)**.** *For each state-action pair* (s, a)*, any value of* $\boldsymbol{\theta}$*, and all parameter components* i, j *there exist constants* $0 \leq G, F < \infty$ *such that:*

$$\left|\nabla_{\theta_i} \log \pi_{\boldsymbol{\theta}}(a|s)\right| \leq G, \qquad \left|\frac{\partial^2}{\partial\theta_i \partial\theta_j} \log \pi_{\boldsymbol{\theta}}(a|s)\right| \leq F.$$

2) FG Approximation. Since we cannot compute an exact full gradient, we require the variance of the estimator to be bounded. This assumption is similar in spirit to the one in (Harikandeh et al., 2015).

Assumption 4.2 (On the variance of the gradient estimator)**.** *There is a constant* $V < \infty$ *such that, for any policy* $\pi_{\boldsymbol{\theta}}$*:*

$$\mathbb{V}\mathrm{ar}\left[g(\cdot|\boldsymbol{\theta})\right] \leq V.$$

3) Non-stationarity. Similarly to what is done in SL (Cortes et al., 2010), we require the variance of the importance weight to be bounded.

[6] For any vector $\mathbf{x}$, we use $\mathbb{V}\mathrm{ar}[\mathbf{x}]$ to denote the trace of the covariance matrix, i.e., $\mathrm{Tr}(\mathbb{E}\left[(\mathbf{x} - \mathbb{E}\left[\mathbf{x}\right])(\mathbf{x} - \mathbb{E}\left[\mathbf{x}\right])^T\right])$.

Assumption 4.3 (On the variance of importance weights)**.** *There is a constant* $W < \infty$ *such that, for each pair of policies encountered in Algorithm 2 and for each trajectory,*

$$\mathbb{V}ar\left[\omega(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)\right] \leq W, \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d, \tau \sim p(\cdot|\boldsymbol{\theta}_1).$$

Differently from Assumptions 4.1 and 4.2, Assumption 4.3 must be enforced by a proper handling of the epoch size m.

We can now state the convergence guarantees for SVRPG.

Theorem 4.4 (Convergence of the SVRPG algorithm)**.** *Assume the REINFORCE or the G(PO)MDP gradient estimator is used in SVRPG (see Equation* (4)*). Under Assumptions 4.1, 4.2 and 4.3, the parameter vector* $\boldsymbol{\theta}_A$ *returned by Algorithm 2 after* $T = m \times S$ *iterations has, for some positive constants* ψ, ζ, ξ *and for proper choice of the step size* α *and the epoch size* m*, the following property:*

$$\mathbb{E}\left[\|\nabla J(\boldsymbol{\theta}_A)\|_2^2\right] \leq \frac{J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}_0)}{\psi T} + \frac{\zeta}{N} + \frac{\xi}{B},$$

where $\boldsymbol{\theta}^*$ *is a global optimum and* ψ, ζ, ξ *depend only on* G, F, V, W, α *and* m*.*

Refer to Appendix B for a detailed proof involving the definition of the constants and the meta-parameter constraints. By analysing the upper-bound in Theorem 4.4 we observe that: I) the $O(1/T)$ term is coherent with results on non-concave SVRG (e.g., Reddi et al., 2016a); II) the $O(1/N)$ term is due to the FG approximation and is analogous to the one in (Harikandeh et al., 2015); III) the $O(1/B)$ term is due to importance weighting. To achieve asymptotic convergence, the batch size N and the mini-batch size B should increase over time. In practice, it is enough to choose N and B large enough to make the second and the third term negligible, i.e., to mitigate the variance introduced by FG approximation and importance sampling, respectively. Once the last two terms can be neglected, the number of trajectories needed to achieve $\|\nabla J(\boldsymbol{\theta})\|_2^2 \leq \epsilon$ is $O(\frac{B+N/m}{\epsilon})$. In this sense, an advantage over batch gradient ascent can be achieved with properly selected meta-parameters. In Section 5.2 we propose a joint selection of step size α and epoch size m. Finally, from the return statement of Algorithm 2, it is worth noting that $J(\boldsymbol{\theta}_A)$ can be seen as the average performance of all the policies tried by the algorithm. This is particularly meaningful in the context of online learning that we are considering in this paper.

5. Remarks on SVRPG

The convergence guarantees presented in the previous section come with requirements on the meta-parameters (i.e., α and m) that may be too conservative for practical applications. Here we provide a practical and automatic way to choose the step size α and the number of sub-iterations m performed between snapshots. Additionally, we provide

a variant of SVRPG exploiting a variance-reduction technique for importance weights. Despite lacking theoretical guarantees, we will show in Section 7 that this method can outperform the baseline SVRPG (Algorithm 2).

5.1. Full Gradient Update

As noted in Section 3, the update performed at the beginning of each epoch is equivalent to a full-gradient update. In our setting, where collecting samples is particularly expensive, the B trajectories collected using the snapshot trajectory $\pi_{\widetilde{\boldsymbol{\theta}}^s}$ feels like a waste of data (the term $\sum_i g(\tau_i) - \omega(\tau_i)g(\tau_i) = 0$ since $\boldsymbol{\theta}_0 = \widetilde{\boldsymbol{\theta}}$). In practice, we just perform an approximate full gradient update using the N trajectories sampled to compute $\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)$, i.e.,

$$\boldsymbol{\theta}_1^{s+1} = \widetilde{\boldsymbol{\theta}}^s + \alpha \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)$$
$$\boldsymbol{\theta}_{t+1}^{s+1} = \boldsymbol{\theta}_t^{s+1} + \alpha \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}) \text{ for } t = 1, \dots, m-1.$$

In the following, we will always use this practical variant.

5.2. Meta-Parameter Selection

The step size α is crucial to balance variance reduction and efficiency, while the epoch length m influences the variance introduced by the importance weights. Low values of m are associated with small variance but increase the frequency of snapshot points (which means many FG computations). High values of m may move policy $\pi_{\boldsymbol{\theta}_t}$ far away from the snapshot policy $\pi_{\widetilde{\boldsymbol{\theta}}}$, causing large variance in the importance weights. We will jointly set the two meta-parameters.

Adaptive step size. A standard way to deal with noisy gradients is to use adaptive strategies to compute the step size. ADAptive Moment estimation (ADAM) (Kingma & Ba, 2014) stabilizes the parameter update by computing learning rates for each parameter based on an incremental estimate of the gradient variance. Due to this feature, we would like to incorporate ADAM in the structure of the SVRPG update. Recall that SVRPG performs two different updates of the parameters $\boldsymbol{\theta}$: I) FG update in the snapshot; II) corrected gradient update in the sub-iterations. Given this structure, we suggest using two separate ADAM estimators:

$$\boldsymbol{\theta}_1^{s+1} = \widetilde{\boldsymbol{\theta}}^s + \alpha_s^{\text{FG}} \left(\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) \right)$$
$$\boldsymbol{\theta}_{t+1}^{s+1} = \boldsymbol{\theta}_t^{s+1} + \alpha_{s+1,t}^{\text{SI}} \left(\blacktriangledown J(\boldsymbol{\theta}_t^{s+1}) \right) \text{ for } t = 1, \dots, m-1,$$

where α_s^{FG} is associated with the snapshot and $\alpha_{s+1,t}^{\text{SI}}$ with the sub-iterations (see Appendix D for details). By doing so, we decouple the contribution of the variance due to the approximate FG from the one introduced by the sub-iterations. Note that these two terms have different orders of magnitude since are estimated with a different number of trajectories ($B \ll N$) and the estimator in the snapshot does not require importance weights. The use of two ADAM estimators allows to capture and exploit this property.

Adaptive epoch length. It is easy to imagine that a predefined schedule (e.g., m fixed in advance or changed with a policy-independent process) may poorly perform due to the high variability of the updates. In particular, given a fixed number of sub-iterations m, the variance of the updates in the sub-iterations depends on the snapshot policy and the sampled trajectories. Since the ADAM estimate partly captures such variability, we propose to take a new snapshot (i.e., interrupt the sub-iterations) whenever the step size α^{SI} proposed by ADAM for the sub-iterations is smaller than the one for the FG (i.e., α^{FG}). If the latter condition is verified, it amounts to say that the noise in the corrected gradient has overcome the information of the FG. Formally, the stopping condition is as follows

$$\textbf{If } \frac{\alpha^{\text{FG}}}{N} > \frac{\alpha^{\text{SI}}}{B} \textbf{ then} \text{ take snapshot,}$$

where we have introduced N and B to take into account the trajectory efficiency (i.e., weighted advantage). The less the number of trajectories used to update the policy, the better. Including the batch sizes in the stopping condition allows us to optimize the trade-off between the quality of the updates and the cost of performing them.

5.3. Normalized Importance Sampling

As mentioned in Section 5.2, importance weights are an additional source of variance. A standard way to cope with this issue is self-normalization (e.g., Precup, 2000; Owen, 2013). This technique can reduce the variance of the importance weights at the cost of introducing some bias (Owen, 2013, Chapter 9). Whether the trade-off is advantageous depends on the specific task. Introducing self-normalization in the context of our algorithm, we switch from Eq. (4) to:

$$\blacktriangledown J(\boldsymbol{\theta}_t) = \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}) + \frac{1}{B} \sum_{i=0}^{B-1} [g(\tau_i|\boldsymbol{\theta}_t)] - \frac{1}{\Omega} \sum_{i=0}^{B-1} \left[\omega(\tau_i|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}}) g(\tau_i|\widetilde{\boldsymbol{\theta}}) \right].$$

where $\Omega = \sum_{i=0}^{B-1} \omega(\tau_i|\boldsymbol{\theta}_t, \widetilde{\boldsymbol{\theta}})$. In Section 7 we show that self-normalization can provide a performance improvement.

6. Related Work

Despite the considerable interest received in SL, variance-reduced gradient approaches have not attracted the RL community. As far as we know, there are just two applications of SVRG in RL. The first approach (Du et al., 2017) aims to exploit SVRG for policy evaluation. The policy evaluation problem is more straightforward than the one faced in this paper (control problem). In particular, since the goal is to evaluate just the performance of a predefined policy, the optimization problem is stationary. The setting considered in the paper is the one of policy evaluation by minimizing the

empirical mean squared projected Bellman error (MSPBE) with a linear approximation of the value function. Du et al. (2017) shown that this problem can be equivalently reformulated as a convex-concave saddle-point problem that is characterized by a finite-sum structure. This problem can be solved using a variant of SVRG (Palaniappan & Bach, 2016) for which convergence guarantees have been provided. The second approach (Xu et al., 2017) uses SVRG as a practical method to solve the optimization problem faced by Trust Region Policy Optimization (TRPO) at each iteration. This is just a direct application of SVRG to a problem having finite-sum structure since no specific structure of the RL problem is exploited. It is worth to mention that, for practical reasons, the authors proposed to use a Newton conjugate gradient method with SVRG.

In the recent past, there has been a surge of studies investigating variance reduction techniques for policy gradient methods. The specific structure of the policy gradient allows incorporating a baseline (i.e., a function $b : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$) without affecting the unbiasedness of the gradient (e.g., Williams, 1992; Weaver & Tao, 2001; Peters & Schaal, 2008b; Thomas & Brunskill, 2017; Wu et al., 2018). Although the baseline can be arbitrarily selected, literature often refers to the optimal baseline as the one minimizing the variance of the estimate. Nevertheless, even the baseline needs to be estimated from data. This fact may partially reduce its effectiveness by introducing variance. Even if these approaches share the same goal as SVRG, they are substantially different. In particular, the proposed SVRPG does not make explicit use of the structure of the policy gradient framework, and it is independent of the underlying gradient estimate (i.e., with or without baseline). This suggests that would be possible to integrate an ad-hoc SVRPG baseline to further reduce the variance of the estimate. Since this paper is about the applicability of SVRG technique to RL, we consider this topic as future work. Additionally, the experiments show that SVRPG has an advantage over G(PO)MPD even when the baseline is used (see the half-cheetah domain in Section 7).

Concerning importance weighting techniques, RL has made extensive use of them for off-policy problems (e.g., Precup, 2000; Thomas et al., 2015). However, as mentioned before, SVRPG cannot be compared to such methods since it is in all respects an on-policy algorithm. Here, importance weighting is just a statistical tool used to preserve the unbiasedness of the corrected gradient.

7. Experiments

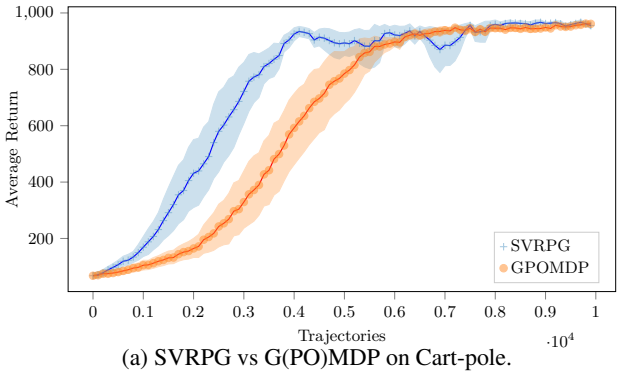
In this section, we evaluate the performance of SVRPG and compare it with policy gradient (PG) on well known continuous RL tasks: Cart-pole balancing and Swimmer (e.g., Duan et al., 2016). We consider G(PO)MDP since it has a smaller variance than REINFORCE. For our algorithm, we use a batch size $N = 100$, a mini-batch size $B = 10$, and the jointly adaptive step size α and epoch length m proposed in Section 5.2. Since the aim of this comparison is to show the improvement that SVRG-flavored variance reduction brings to SG in the policy gradient framework, we set the batch size of the baseline policy gradient algorithm to B. In this sense, we measure the improvement yielded by computing snapshot gradients and using them to adjust parameter updates. Since we evaluate on-line performance over the number of sampled trajectories, the cost of computing such snapshot gradients is automatically taken into consideration. To make the comparison fair, we also use Adam in the baseline PG algorithm, which we will denote simply as G(PO)MDP in the following. In all the experiments, we use deep Gaussian policies with adaptive standard deviation (details on network architecture in Appendix E). Each experiment is run 10 times with a random policy initialization and seed, but this initialization is shared among the algorithms under comparison. The length of the experiment, i.e., the total number of trajectories, is fixed for each task. Performance is evaluated by using test-trajectories on a subset of the policies considered during the learning process. We provide average performance with 90% bootstrap confidence intervals. Task implementations are from the *rllab* library (Duan et al., 2016), on which our agents are also based.[7] More details on meta-parameters and exhaustive task descriptions are provided in Appendix E.

Figure 1a compares SVRPG with G(PO)MDP on a continuous variant of the classical Cart-pole task, which is a 2D balancing task. Despite using more trajectories on average for each parameter update, our algorithm shows faster convergence, which can be ascribed to the better quality of updates due to variance reduction.

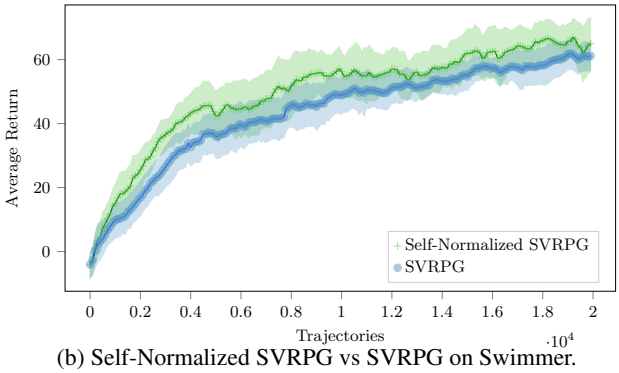
The Swimmer task is a 3D continuous-control locomotion task. This task is more difficult than cart-pole. In particular, the longer horizon and the more complex dynamics can have a dangerous impact on the variance of importance weights. In this case, the self-normalization technique proposed in Section 5.3 brings an improvement (even if not statistically significant), as shown in Figure 1b. Figure 1c shows self-normalized SVRPG against G(PO)MDP. Our algorithm outperforms G(PO)MDP for almost the entire learning process. Also here, we note an increase of speed in early iterations, and, toward the end of the learning process, the improvement becomes statistically significant.

Preliminary results on actor-critic. Another variance-reduction technique in policy gradient consists of using baselines or *critics*. This tool is orthogonal to the methods described in this paper, and the theoretical results of Section 4 are general in this sense. In the experiments de-

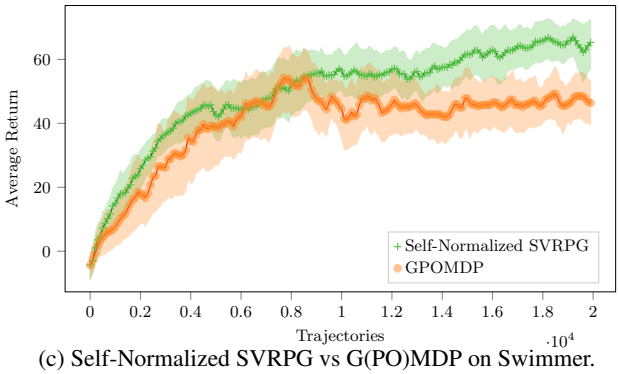
[7] Code available at `github.com/Dam930/rllab`.



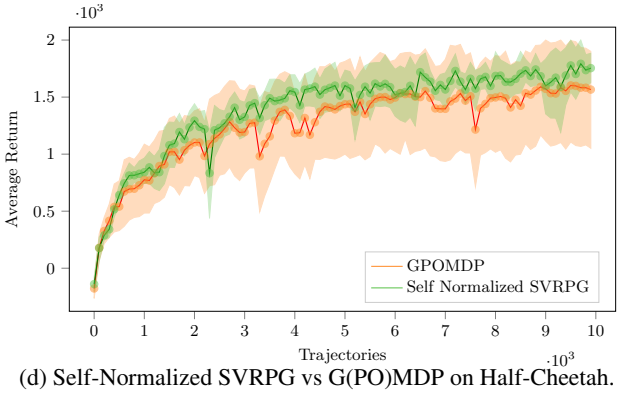
(a) SVRPG vs G(PO)MDP on Cart-pole.



(b) Self-Normalized SVRPG vs SVRPG on Swimmer.



(c) Self-Normalized SVRPG vs G(PO)MDP on Swimmer.



(d) Self-Normalized SVRPG vs G(PO)MDP on Half-Cheetah.

Figure 1: Comparison of on-line performance over sampled trajectories, with 90% confidence intervals.

scribed so far, we compared against the so-called *actor-only* G(PO)MDP, i.e., without the baseline. To move towards a more general understanding of the variance issue in policy gradient, we also test SVRPG in an *actor-critic* scenario. To do so, we consider the more challenging MuJoCo (Todorov et al., 2012) Half-cheetah task, a 3D locomotion task that has a larger state-action space than Swimmer. Figure 1d compares self-normalized SVRPG and G(PO)MDP on Half-cheetah, using the critic suggested in (Duan et al., 2016) for both algorithms. Results are promising, showing that a combination of the baseline usage and SVRG-like variance reduction can yield an improvement that the two techniques alone are not able to achieve. Moreover, SVRPG presents a noticeably lower variance. The performance of actor-critic G(PO)MDP[8] on Half-Cheetah is coherent with the one reported in (Duan et al., 2016). Other results are not comparable since we did not use the critic.

8. Conclusion

In this paper, we introduced SVRPG, a variant of SVRG designed explicitly for RL problems. The control problem considered in the paper has a series of difficulties that are not common in SL. Among them, non-concavity and approximate estimates of the FG have been analysed independently in SL (e.g., Allen-Zhu & Hazan, 2016; Reddi et al., 2016a; Harikandeh et al., 2015) but never combined. Nevertheless, the main issue in RL is the non-stationarity of the sampling process since the distribution underlying the objective function is policy-dependent. We have shown that by exploiting importance weighting techniques, it is possible to overcome this issue and preserve the unbiasedness of the corrected gradient. We have additionally shown that, under mild assumptions that are often verified in RL applications, it is possible to derive convergence guarantees for SVRPG. Finally, we have empirically shown that practical variants of the theoretical SVRPG version can outperform classical actor-only approaches on benchmark tasks. Preliminary results support the effectiveness of SVRPG also with a commonly used baseline for the policy gradient. Despite that, we believe that it will be possible to derive a baseline designed explicitly for SVRPG to exploit the RL structure and the SVRG idea jointly. Another possible improvement would be to employ the natural gradient (Kakade, 2002) to better control the effects of parameter updates on the variance of importance weights. Future work should also focus on making batch sizes N and B adaptive, as suggested in (Papini et al., 2017).

[8] Duan et al. (2016) report results on REINFORCE. However, inspection on *rllab* code and documentation reveals that it is actually PGT (Sutton et al., 2000), which is equivalent to G(PO)MDP (shown by Peters & Schaal, 2008b). Using the name REINFORCE in a general way is inaccurate, but widespread.

Acknowledgments

This research was supported in part by French Ministry of Higher Education and Research, Nord-Pas-de-Calais Regional Council and French National Research Agency (ANR) under project ExTra-Learn (n.ANR-14-CE24-0010-01).

References

Allen-Zhu, Zeyuan and Hazan, Elad. Variance reduction for faster non-convex optimization. In *International Conference on Machine Learning*, pp. 699–707, 2016.

Baxter, Jonathan and Bartlett, Peter L. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

Bietti, Alberto and Mairal, Julien. Stochastic optimization with variance reduction for infinite datasets with finite sum structure. In *Advances in Neural Information Processing Systems*, pp. 1622–1632, 2017.

Bottou, Léon and LeCun, Yann. Large scale online learning. In *Advances in neural information processing systems*, pp. 217–224, 2004.

Cauchy, Augustin. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris*, 25(1847):536–538, 1847.

Cortes, Corinna, Mansour, Yishay, and Mohri, Mehryar. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pp. 442–450, 2010.

Defazio, Aaron, Bach, Francis, and Lacoste-Julien, Simon. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in Neural Information Processing Systems*, pp. 1646–1654, 2014a.

Defazio, Aaron, Domke, Justin, et al. Finito: A faster, permutable incremental gradient method for big data problems. In *International Conference on Machine Learning*, pp. 1125–1133, 2014b.

Du, Simon S., Chen, Jianshu, Li, Lihong, Xiao, Lin, and Zhou, Dengyong. Stochastic variance reduction methods for policy evaluation. In *ICML*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1049–1058. PMLR, 2017.

Duan, Yan, Chen, Xi, Houthooft, Rein, Schulman, John, and Abbeel, Pieter. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pp. 1329–1338, 2016.

Furmston, Thomas and Barber, David. A unifying perspective of parametric policy search methods for markov decision processes. In *NIPS*, pp. 2726–2734, 2012.

Ghadimi, Saeed and Lan, Guanghui. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

Harikandeh, Reza, Ahmed, Mohamed Osama, Virani, Alim, Schmidt, Mark, Konečnỳ, Jakub, and Sallinen, Scott. Stopwasting my gradients: Practical svrg. In *Advances in Neural Information Processing Systems*, pp. 2251–2259, 2015.

Johnson, Rie and Zhang, Tong. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pp. 315–323, 2013.

Jurčíček, Filip. Reinforcement learning for spoken dialogue systems using off-policy natural gradient method. In *Spoken Language Technology Workshop (SLT), 2012 IEEE*, pp. 7–12. IEEE, 2012.

Kakade, Sham M. A natural policy gradient. In *Advances in neural information processing systems*, pp. 1531–1538, 2002.

Kingma, Diederik P and Ba, Jimmy. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Konečnỳ, Jakub, Liu, Jie, Richtárik, Peter, and Takáč, Martin. Mini-batch semi-stochastic gradient descent in the proximal setting. *IEEE Journal of Selected Topics in Signal Processing*, 10(2):242–255, 2016.

Mairal, Julien. Incremental majorization-minimization optimization with application to large-scale machine learning. *SIAM Journal on Optimization*, 25(2):829–855, 2015.

Nemirovskii, Arkadii, Yudin, David Borisovich, and Dawson, Edgar Ronald. Problem complexity and method efficiency in optimization. 1983.

Nesterov, Yurii. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

Owen, Art B. *Monte Carlo theory, methods and examples*. 2013.

Palaniappan, Balamurugan and Bach, Francis. Stochastic variance reduction methods for saddle-point problems. In *NIPS*, pp. 1408–1416, 2016.

Papini, Matteo, Pirotta, Matteo, and Restelli, Marcello. Adaptive batch size for safe policy gradients. In *Advances in Neural Information Processing Systems*, pp. 3594–3603, 2017.

Peters, Jan and Schaal, Stefan. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21 (4):682–697, 2008a.

Peters, Jan and Schaal, Stefan. Reinforcement learning of motor skills with policy gradients. *Neural networks*, 21 (4):682–697, 2008b.

Pirotta, Matteo, Restelli, Marcello, and Bascetta, Luca. Adaptive step-size for policy gradient methods. In *Advances in Neural Information Processing Systems*, pp. 1394–1402, 2013.

Pirotta, Matteo, Restelli, Marcello, and Bascetta, Luca. Policy gradient in lipschitz markov decision processes. *Machine Learning*, 100(2):255–283, 2015. ISSN 1573-0565. doi: 10.1007/s10994-015-5484-1.

Precup, Doina. Eligibility traces for off-policy policy evaluation. *Computer Science Department Faculty Publication Series*, pp. 80, 2000.

Reddi, Sashank J, Hefny, Ahmed, Sra, Suvrit, Poczos, Barnabas, and Smola, Alex. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pp. 314–323, 2016a.

Reddi, Sashank J, Sra, Suvrit, Póczos, Barnabás, and Smola, Alex. Fast incremental method for nonconvex optimization. *arXiv preprint arXiv:1603.06159*, 2016b.

Robbins, Herbert and Monro, Sutton. A stochastic approximation method. *The annals of mathematical statistics*, pp. 400–407, 1951.

Roux, Nicolas L, Schmidt, Mark, and Bach, Francis R. A stochastic gradient method with an exponential convergence _rate for finite training sets. In *Advances in Neural Information Processing Systems*, pp. 2663–2671, 2012.

Rubinstein, Reuven Y Reuven Y. Simulation and the monte carlo method. Technical report, 1981.

Sutton, Richard S and Barto, Andrew G. *Reinforcement learning: An introduction*. MIT press Cambridge, 1998.

Sutton, Richard S, McAllester, David A, Singh, Satinder P, and Mansour, Yishay. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pp. 1057–1063, 2000.

Thomas, Philip S. and Brunskill, Emma. Policy gradient methods for reinforcement learning with function approximation and action-dependent baselines. *CoRR*, abs/1706.06643, 2017.

Thomas, Philip S, Theocharous, Georgios, and Ghavamzadeh, Mohammad. High-confidence off-policy evaluation. In *AAAI*, pp. 3000–3006, 2015.

Todorov, Emanuel, Erez, Tom, and Tassa, Yuval. Mujoco: A physics engine for model-based control. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pp. 5026–5033. IEEE, 2012.

Weaver, Lex and Tao, Nigel. The optimal reward baseline for gradient-based reinforcement learning. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pp. 538–545. Morgan Kaufmann Publishers Inc., 2001.

Williams, Ronald J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Wu, Cathy, Rajeswaran, Aravind, Duan, Yan, Kumar, Vikash, Bayen, Alexandre M, Kakade, Sham, Mordatch, Igor, and Abbeel, Pieter. Variance reduction for policy gradient with action-dependent factorized baselines. *International Conference on Learning Representations*, 2018. accepted as oral presentation.

Xu, Tianbing, Liu, Qiang, and Peng, Jian. Stochastic variance reduction for policy gradient estimation. *CoRR*, abs/1710.06034, 2017.

Zhao, Tingting, Hachiya, Hirotaka, Niu, Gang, and Sugiyama, Masashi. Analysis and improvement of policy gradient estimation. In *Advances in Neural Information Processing Systems*, pp. 262–270, 2011.

A. Policy Gradient Estimators

We present a brief overview of the two most widespread gradient estimators (REINFORCE (Williams, 1992) and G(PO)MDP (Baxter & Bartlett, 2001)) both in on-policy and off-policy settings. Let $\tau = \{\langle s_t, a_t\rangle\}_{t=0}^{H} = \{z_t\}_{t=0}^{H} = z_{0:H}$ is a $(H+1)$-steps trajectory. Given that τ depends on the MDP $M = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \rho\}$ and the actual policy π, the trajectory is said drawn from density distribution $p(\tau|\pi, M)$ defined as:

$$p(\tau|\pi, M) = \rho(s_0)\pi(z_0)\prod_{k=1}^{H}\mathcal{P}(s_k|z_{k-1})\pi(z_k).$$

We can now recall the definition of policy performance

$$J(\pi_{\boldsymbol{\theta}}) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi_{\boldsymbol{\theta}})}\left[\mathcal{R}(\tau)|M\right],$$

where $\mathcal{R}(\tau) = \sum_{t=0}^{H}\gamma^t\mathcal{R}(z_t)$. The policy gradient $\nabla J(\boldsymbol{\theta})$ is

$$\nabla J(\boldsymbol{\theta}) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi_{\boldsymbol{\theta}})}\left[\nabla\log p(\tau|\boldsymbol{\theta})\mathcal{R}(\tau)\right] = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi_{\boldsymbol{\theta}})}\left[\sum_{j=0}^{H}\gamma^j\mathcal{R}(z_j)\sum_{i=0}^{H}\nabla\log\pi(z_i)\right]. \tag{7}$$

On-policy setting. Consider a policy $\pi_{\boldsymbol{\theta}}$ and let $\mathcal{D}_N = \{\tau_i\}_{i=1}^{N}$ be a dataset collected using policy $\pi_{\boldsymbol{\theta}}$. The REINFORCE gradient estimator (Williams, 1992) provides a simple, unbiased way of estimating the gradient:

$$\widehat{\nabla}_N J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\underbrace{\left(\sum_{h=0}^{H}\nabla\log\pi_{\boldsymbol{\theta}}(z_h^n)\right)\left(\sum_{h=0}^{H}\gamma^h r_h^n - b(z_h^n)\right)}_{g(\tau_n|\boldsymbol{\theta}):=\nabla\log p(\tau_n|\boldsymbol{\theta})\mathcal{R}(\tau_n)},$$

where subscripts denote the time step, superscripts denote the trajectory, r_h^n is the reward actually collected at time h from trajectory τ^n and $b : \mathcal{S}\times\mathcal{A}\to\mathbb{R}$ (e.g., Thomas & Brunskill, 2017). The G(PO)MDP gradient estimator (Baxter & Bartlett, 2001) is a refinement of REINFORCE which is subject to less variance (Zhao et al., 2011) while preserving the unbiasedness:

$$\widehat{\nabla}_N J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{n=1}^{N}\underbrace{\sum_{h=0}^{H}\left(\sum_{k=0}^{h}\nabla\log\pi_{\boldsymbol{\theta}}(z_h^n)\right)\left(\gamma^h r_h^n - b(z_h^n)\right)}_{g(\tau_n|\boldsymbol{\theta})}.$$

G(PO)MDP can be seen as a more efficient implementation of the REINFORCE algorithm. In fact, the latter does not perform an optimal credit assignment since it ignores that the reward at time t does not depend on the action performed after time t. G(PO)MDP overcomes this issue taking into account the causality of rewards in the REINFORCE definition of policy gradient.

Off-policy setting. In off-policy setting two policies, called behavioural π^B and target π^T, are involved. The first is used to select actions for the interaction with the system, while the second is used to evaluate the agent performance and it is improved in each update. Suppose now that we aim to estimate the performance of the target policy π^T but we have samples collected using policy π^B. We can use importance weight correction to correct the shift in the distribution and obtain an unbiased estimate of $J(\pi^T)$:

$$J(\pi^T) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^T)}\left[\mathcal{R}(\tau)\right] = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^B)}\left[\omega(\tau)\mathcal{R}(\tau)\right]$$

where $\omega(\tau|\pi^B, \pi^T) = \frac{p(\tau,\pi^T)}{p(\tau|\pi^B)} = \omega(z_{0:H}|\pi^B, \pi^T) = \prod_{w=0}^{H}\omega(z_w|\pi^B, \pi^T)$ and $\omega(z_w|\pi^B, \pi^T) = \frac{\pi^T(a_w|s_w)}{\pi^B(a_w|s_w)}$

The definition of the off-policy version of (7) is (e.g., Jurčíček, 2012)

$$\nabla J(\pi^T) = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^B)}\left[\omega(\tau|\pi^B, \pi^T)\nabla\log p(\tau|\pi^T)\mathcal{R}(\tau)\right] = \mathop{\mathbb{E}}_{\tau\sim p(\cdot|\pi^B)}\left[\omega(\tau|\pi^B, \pi^T)g(\tau|\pi^T)\right]. \tag{8}$$

For $\omega(\tau|\pi^B, \pi^T)$ being well defined the behavioural policy should have non-zero probability of selecting any action in every state i.e., $\pi^B(a|s) > 0$ for any $(s, a) \in \mathcal{S} \times \mathcal{A}$. Equation 8 is important for proving Theorem 4.4 since it provides a common representation of REINFORCE and G(PO)MDP.

The off-policy version of REINFORCE is easily obtained by taking the empirical average of (8):

$$\nabla J(\pi^T) = \frac{1}{N} \sum_{n=1}^{N} \omega(\tau^n|\pi^B, \pi^T) \underbrace{\left(\sum_{h=0}^{H} \nabla \log \pi^T(z_h^n) \right) \left(\sum_{h=0}^{H} \gamma^h \mathcal{R}(z_h^n) \right)}_{g(\tau_n|\pi^T)}.$$

The G(PO)MDP off-policy estimator is defined as follows

$$\nabla J(\pi^T) = \frac{1}{N} \sum_{n=1}^{N} \underbrace{\sum_{h=0}^{H} \left(\sum_{k=0}^{h} \nabla \log \pi^T(z_k^n) \right) \gamma^h \mathcal{R}(z_h^n) \omega(z_{0:h}|\pi^B, \pi^T)}_{\omega(\tau_n|\pi^B, \pi^T) g(\tau_n|\pi^T)}.$$

B. Proofs

In this section, we prove all the claims made in the paper, with the primary objective of proving Theorem 4.4. Our proof is adapted from the one of Theorem 2 from (Reddi et al., 2016a) and has a very similar structure, but with all the additional challenges and assumptions described in Section 4.

Note that in the following we will make wide use of the following properties.

Assumption B.1. *We consider an estimate $\widehat{\nabla}_N J(\boldsymbol{\theta})$ as in Eq. 2 such that*

1. *On-policy Unbiased Estimator.*

$$\mathop{\mathbb{E}}_{\tau_i \sim \pi(\cdot|\boldsymbol{\theta})} \left[\widehat{\nabla}_N J(\boldsymbol{\theta}) \right] = \mathop{\mathbb{E}}_{\tau_i \sim \pi(\cdot|\boldsymbol{\theta})} \left[\frac{1}{N} \sum_{i=0}^{N} g(\tau_i|\boldsymbol{\theta}) \right] = \nabla J(\boldsymbol{\theta})$$

2. *Off-policy Unbiased Estimator.*

$$\mathop{\mathbb{E}}_{\tau_i \sim \Delta(\cdot|\pi^B)} \left[\frac{1}{N} \sum_{i=0}^{N} \omega(\tau_i|\pi^B, \boldsymbol{\theta}) g(\tau_i|\boldsymbol{\theta}) \right] = \nabla J(\boldsymbol{\theta})$$

Note that these assumptions are verified by REINFORCE and G(PO)MDP.

Definitions

We give some additional definitions which will be useful in the proofs.

Definition B.1. For a random variable X:

$$\mathbb{E}_s [X] = \mathop{\mathbb{E}}_{\tau_j \sim \pi(\cdot|\widetilde{\boldsymbol{\theta}}^s) \forall j \in \mathbf{N}} \left[X|\widetilde{\boldsymbol{\theta}}^s \right],$$

where $\widetilde{\boldsymbol{\theta}}^s$ is defined in Algorithm 2 and $\mathbf{N} = [0, \dots, N)$.

We introduce the notation $\tau_{i,h}$ to denote the h-th trajectory collected using policy $\boldsymbol{\theta}_i^{s+1}$ where s will be clear from the context.

Definition B.2. For a random variable X:

$$\begin{aligned}\mathbb{E}_{t|s}[X] &\coloneqq \mathop{\mathbb{E}}_{\substack{\tau_j\sim\pi(\cdot|\widetilde{\boldsymbol{\theta}}^s)\forall j\in\mathbf{N}\\ \tau_{i,h}\sim\pi(\cdot|\boldsymbol{\theta}_i^{s+1})\forall h\in\mathbf{B},\text{ for } i=0,\dots,t}}\left[X|\widetilde{\boldsymbol{\theta}^s}\right]\\ &\coloneqq \mathop{\mathbb{E}}_{\tau_j\sim\pi(\cdot|\widetilde{\boldsymbol{\theta}}^s)\forall j\in\mathbf{N}}\left[\mathop{\mathbb{E}}_{\tau_{0,h}\sim\pi(\cdot|\boldsymbol{\theta}_0^{s+1})\forall h\in\mathbf{B}}\left[\dots\mathop{\mathbb{E}}_{\tau_{t,h}\sim\pi(\cdot|\boldsymbol{\theta}_t^{s+1})\forall h\in\mathbf{B}}\left[X|\boldsymbol{\theta}_t^{s+1}\right]\dots|\boldsymbol{\theta}_0^{s+1}\right]|\widetilde{\boldsymbol{\theta}}^s\right]\\ &= \mathop{\mathbb{E}}_{t-1|s}\left[\mathop{\mathbb{E}}_{\tau_{t,h}\sim\pi(\cdot|\boldsymbol{\theta}_t^{s+1})}\left[X|\boldsymbol{\theta}_t^{s+1}\right]\right]\end{aligned}$$

where the sequence $\widetilde{\boldsymbol{\theta}}^s, \boldsymbol{\theta}_0^{s+1}, \dots, \boldsymbol{\theta}_t^{s+1}$ is defined in Algorithm 2, $\mathbf{N} = [0, \dots, N)$, and $\mathbf{B} = [0, \dots, B)$. To avoid inconsistencies, we also define $\mathbb{E}_{(-1)|s}[X] \coloneqq \mathbb{E}_s[X]$.

Intuitively, the $\mathbb{E}_{t|s}[\cdot]$ operator computes the expected value with respect to the sampling of trajectories from the snapshot $\widetilde{\boldsymbol{\theta}}^s$ up to the t-th iteration included. Note that the order in which expected values are taken is important since each $\boldsymbol{\theta}_t^{s+1}$ is function of previously sampled trajectories and is used to sample new ones.

Definition B.3. For random vectors X, Y:

$$\begin{aligned}\mathbb{C}\text{ov}_s(X,Y) &\coloneqq \text{Tr}\left(\mathbb{E}_{t|s}\left[(X-\mathbb{E}_{t|s}[X])(Y-\mathbb{E}_{t|s}[Y])^T\right]\right),\\ \mathbb{V}\text{ar}_s[X] &\coloneqq \mathbb{C}\text{ov}_s(X,Y),\end{aligned}$$

where $\text{Tr}(\cdot)$ denotes the trace of a matrix. From the linearity of expected value we have the following:

$$\mathbb{V}\text{ar}_s[X] = \mathbb{E}_s\left[\|X-\mathbb{E}_s[X]\|^2\right] \tag{9}$$

$\mathbb{C}\text{ov}_{t|s}(X,Y)$ and $\mathbb{V}\text{ar}_{t|s}[X]$ are defined in the same way from $\mathbb{E}_{t|s}[X]$.

Definition B.4. The full gradient estimation error is:

$$e_s \coloneqq \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) - \nabla J(\widetilde{\boldsymbol{\theta}}^s)$$

Definition B.5. The ideal SVRPG gradient estimate is:

$$\begin{aligned}\mathbb{\nabla} J(\boldsymbol{\theta}_t^{s+1}) &\coloneqq \nabla J(\widetilde{\boldsymbol{\theta}}^s) + g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i|\boldsymbol{\theta}_t^{s+1}, \widetilde{\boldsymbol{\theta}}^s) g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\\ &= \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}) - \widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) + \nabla J(\widetilde{\boldsymbol{\theta}}^s)\\ &= \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}) - e_s\end{aligned}$$

Basic Lemmas

We prove two basic properties of the SVRPG update.

Lemma 3.1. *Let $\widehat{\nabla}_N J(\boldsymbol{\theta})$ be an unbiased estimator of (1) and let $\boldsymbol{\theta}^* \in \arg\min_{\boldsymbol{\theta}}\{J(\boldsymbol{\theta})\}$. Then, the SVRG estimate in (4) is* unbiased

$$\mathbb{E}[\blacktriangledown J(\boldsymbol{\theta})] = \nabla J(\boldsymbol{\theta}). \tag{5}$$

and regardless of the mini-batch size B:[9]

$$\mathbb{V}\text{ar}[\blacktriangledown J(\boldsymbol{\theta}^*)] = \mathbb{V}\text{ar}\left[\widehat{\nabla}_N J(\boldsymbol{\theta}^*)\right]. \tag{6}$$

Proof.

$$\begin{aligned}\mathbb{E}[\blacktriangledown J(\boldsymbol{\theta})] &= \mathbb{E}\left[\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}})\right] + \mathbb{E}\left[\widehat{\nabla}_B J(\boldsymbol{\theta})\right] - \mathbb{E}\left[\frac{1}{B}\sum_{i=0}^{B-1}\omega(\tau_i|\boldsymbol{\theta},\widetilde{\boldsymbol{\theta}}) g(\tau_i|\widetilde{\boldsymbol{\theta}})\right]\\ &= \nabla J(\widetilde{\boldsymbol{\theta}}) + \nabla J(\boldsymbol{\theta}) - \nabla J(\widetilde{\boldsymbol{\theta}}) = \nabla J(\boldsymbol{\theta}).\end{aligned}$$

[9] For any vector $\mathbf{x}$, we use $\mathbb{V}\text{ar}[\mathbf{x}]$ to denote the trace of the covariance matrix, i.e., $\text{Tr}(\mathbb{E}\left[(\mathbf{x}-\mathbb{E}[\mathbf{x}])(\mathbf{x}-\mathbb{E}[\mathbf{x}])^T\right])$.

Note that the importance weight is necessary to guarantee unbiasedness, since the τ_i are sampled from $\pi_{\boldsymbol{\theta}}$. As $\boldsymbol{\theta} \to \boldsymbol{\theta}^*$, also $\widetilde{\boldsymbol{\theta}} \to \boldsymbol{\theta}^*$. Hence, by continuity of $J(\boldsymbol{\theta})$:

$$\begin{aligned}\mathbb{V}\mathrm{ar}\left[\blacktriangledown J(\boldsymbol{\theta})\right] &\to \mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_N J(\boldsymbol{\theta}^*)\right] + \frac{1}{B}\mathbb{V}\mathrm{ar}\left[g(\tau|\boldsymbol{\theta}^*) - \cancel{\omega(\tau|\boldsymbol{\theta}^*,\boldsymbol{\theta}^*)g(\tau|\boldsymbol{\theta}^*)}\right] \\ &= \mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_N J(\boldsymbol{\theta}^*)\right].\end{aligned}$$

Note that it is important that the trajectories used in the second and the third term are the same for the variance to vanish. □

Ancillary Lemmas

Before addressing the main convergence theorem, we prove some useful lemmas.

Lemma B.2. *Under Assumption 4.1, $J(\boldsymbol{\theta})$ is L-smooth for some positive Lipschitz constant L_J.*

Proof. By definition of $J(\boldsymbol{\theta})$:

$$\begin{aligned}\frac{\partial^2 J(\boldsymbol{\theta})}{\partial\theta_i\partial\theta_j} &= \int_{\mathcal{T}} \frac{\partial^2}{\partial\theta_i\partial\theta_j} p(\tau|\boldsymbol{\theta})\mathcal{R}(\tau)\,\mathrm{d}\tau \\ &= \int_{\mathcal{T}} p(\tau|\boldsymbol{\theta})\nabla\log p_{\boldsymbol{\theta}}(\tau)\nabla\log p_{\boldsymbol{\theta}}(\tau)^T\mathcal{R}(\tau)\,\mathrm{d}\tau + \int_{\mathcal{T}} \pi_{\boldsymbol{\theta}}(\tau)\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log p(\tau|\boldsymbol{\theta})\mathcal{R}(\tau)\,\mathrm{d}\tau \\ &\leq \sup_{\tau\in\mathcal{T}}\{|\mathcal{R}(\tau)|\}\left(H^2G^2 + HF\right) \qquad (10) \\ &= \frac{1-\gamma^H}{1-\gamma}RH\left(HG^2+F\right),\end{aligned}$$

where 10 is from Assumption 4.1. Since the Hessian is bounded, $J(\boldsymbol{\theta})$ is Lipschitz-smooth. □

Lemma B.3. *Under Assumption 4.1, whether we use the REINFORCE or the G(PO)MDP gradient estimator, $g(\tau|\boldsymbol{\theta})$ is Lipschitz continuous with Lipschitz constant L_g, i.e., for any trajectory $\tau \in \mathcal{T}$:*

$$\left\|g(\tau|\boldsymbol{\theta}) - g(\tau|\boldsymbol{\theta}')\right\|_2^2 \leq L_g\left\|\boldsymbol{\theta} - \boldsymbol{\theta}'\right\|_2^2.$$

Proof. For both REINFORCE and G(PO)MDP, $g(\tau|\boldsymbol{\theta})$ is a linear combination of terms of the kind $\nabla\log\pi_{\boldsymbol{\theta}}(a_t|s_t)\gamma^t r_t$ (Peters & Schaal, 2008b). These terms have bounded gradient from the second inequality of Assumption 4.1 and the fact that $|r_t| \leq R$. If a baseline is used in REINFORCE or G(PO)MDP, we only need the additional assumption that said baseline is bounded. Bounded gradient implies Lipschitz continuity. Finally, the linear combination of Lipschitz continuous functions is Lipschitz continuous. □

Lemma B.4. *Under Assumption 4.1, whether we use the REINFORCE or the G(PO)MDP gradient estimator, for every $\tau \in \mathcal{T}$ and $\boldsymbol{\theta} \in \Theta$, there is a positive constant $\Gamma < \infty$ such that:*

$$\left\|g(\tau|\boldsymbol{\theta})\right\|_2^2 \leq \Gamma.$$

Proof. For REINFORCE we have, from Assumption 4.1:

$$\begin{aligned}\left\|g(\tau|\boldsymbol{\theta})\right\|_2^2 &= \left\|\nabla\log p_{\boldsymbol{\theta}}(\tau)R(\tau)\right\|_2^2 \\ &= \left\|\left(\sum_{t=0}^{H-1}\nabla\log p_{\boldsymbol{\theta}}(a_t|s_t)\right)\left(\sum_{t=0}^{H-1}\gamma^t r_t\right)\right\|_2^2 \leq H^2G^2\frac{(1-\gamma^H)^2}{(1-\gamma)^2}R^2\dim(\boldsymbol{\Theta}) \coloneqq \Gamma\end{aligned}$$

For G(PO)MDP, we do not have a compact expression for g, but since it is derived from REINFORCE by neglecting some terms of the kind $\nabla\log\pi_{\boldsymbol{\theta}}(a_t|s_t)\gamma^t r_t$ (Baxter & Bartlett, 2001; Peters & Schaal, 2008b), the above bound still holds. If a baseline is used in REINFORCE or G(PO)MDP, we only need the additional assumption that said baseline is bounded. □

Lemma B.5. *For any random vector X, the variance (as defined in Definition B.3), can be bounded as follows:*

$$\mathbb{V}ar_s\left[X\right] \leq \mathbb{E}_s\left[\|X\|^2\right].$$

Proof. By using basic properties of expected value and scalar variance:

$$\begin{aligned}\mathbb{V}\text{ar}_s\left[X\right] = \mathbb{E}_s\left[\|X - \mathbb{E}_s\left[X\right]\|^2\right] = \mathbb{E}_s\left[\sum_{i=1}^{\dim(X)}\left(X_i - \mathbb{E}_s\left[X_i\right]\right)^2\right] &= \sum_{i=1}^{\dim(X)}\mathbb{E}_s\left[\left(X_i - \mathbb{E}_s\left[X_i\right]\right)^2\right] \\ &\leq \sum_{i=1}^{\dim(X)}\mathbb{E}_s\left[X_i^2\right] = \mathbb{E}_s\left[\sum_{i=1}^{\dim(X)}X_i^2\right] = \mathbb{E}_s\left[\|X\|^2\right].\end{aligned}$$

□

Lemma B.6. *Under Assumption 4.1 , the expected squared norm of the SVRPG gradient can be bounded as follows:*

$$\mathbb{E}_{t|s}\left[\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|_2^2\right] \leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|_2^2\right] + \frac{L_g^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|_2^2\right] + \frac{1}{N}\mathbb{V}ar_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right] + \frac{\Gamma W}{B}$$

Proof. For ease of notation denote $\omega(\tau_i) := \omega(\tau_i|\boldsymbol{\theta}_t^{s+1}, \widetilde{\boldsymbol{\theta}}^s)$. Then,

$$\begin{aligned}\mathbb{E}_{t|s}\left[\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|_2^2\right] &= \mathbb{E}_{t|s}\left[\left\|\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) + \frac{1}{B}\sum_{i=0}^{B-1}g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \frac{1}{B}\sum_{i=0}^{B-1}\omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right\|^2\right] \\ &= \mathbb{E}_{t|s}\left[\left\|\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) + \frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right) \pm \nabla J(\boldsymbol{\theta}_t^{s+1}) \pm \nabla J(\widetilde{\boldsymbol{\theta}}^s)\right\|^2\right] \\ &\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] + \mathbb{E}_s\left[\left\|\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s) - \mathbb{E}_s\left[\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)\right]\right\|^2\right] \\ &\quad + \mathbb{E}_{t|s}\left[\left\|\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right) - \mathbb{E}_{t|s}\left[\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right)\right]\right\|^2\right] \\ &= \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] + \mathbb{V}\text{ar}_s\left[\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)\right] \\ &\quad + \mathbb{E}_{t|s}\left[\left\|\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right) - \mathbb{E}_{t|s}\left[\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right)\right]\right\|^2\right] \quad (11) \\ &= \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] + \frac{1}{N}\mathbb{V}\text{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right] \\ &\quad + \mathbb{E}_{t|s}\left[\left\|\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right) - \mathbb{E}_{t|s}\left[\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right)\right]\right\|^2\right] \quad (12) \\ &\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|_2^2\right] + \frac{1}{N}\mathbb{V}\text{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right] + \mathbb{E}_{t|s}\left[\left\|\frac{1}{B}\sum_{i=0}^{B-1}\left(g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right)\right\|_2^2\right] \quad (13) \\ &\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|_2^2\right] + \frac{1}{N}\mathbb{V}\text{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right] + \frac{1}{B^2}\sum_{i=0}^{B-1}\mathbb{E}_{t|s}\left[\left\|g(\tau_i|\boldsymbol{\theta}_t^{s+1}) - \omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s) \pm g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right\|_2^2\right] \\ &\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] + \frac{1}{N}\mathbb{V}\text{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]\end{aligned}$$

$$
\begin{aligned}
&+\frac{1}{B^2}\sum_{i=0}^{B-1}\mathbb{E}_{t|s}\left[\left\|g(\tau_i|\boldsymbol{\theta}_t^{s+1})-g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right\|^2\right]+\frac{1}{B^2}\sum_{i=0}^{B-1}\mathbb{E}_{t|s}\left[\left\|g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)-\omega(\tau_i)g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right\|^2\right]\\
&\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{1}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]\\
&\quad+\frac{L_g^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+\frac{1}{B^2}\sum_{i=0}^{B-1}\mathbb{E}_{t|s}\left[\left\|(1-\omega(\tau_i))g(\tau_i|\widetilde{\boldsymbol{\theta}}^s)\right\|^2\right] \qquad (14)\\
&\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{1}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]\\
&\quad+\frac{L_g^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+\Gamma\frac{1}{B^2}\sum_{i=0}^{B-1}\mathbb{E}_{t|s}\left[(\omega(\tau_i)-1)^2\right] \qquad (15)\\
&=\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{1}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{L_g^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+\frac{\Gamma}{B^2}\sum_{i=0}^{B-1}\mathbb{V}\mathrm{ar}_{t|s}\left[\omega(\tau_i)\right]\\
&\leq \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{1}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{L_g^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+\frac{\Gamma W}{B}, \qquad (16)
\end{aligned}
$$

where (11) is from (9), (12) is from the definition of $\widehat{\nabla}_N J(\boldsymbol{\theta})$, (13) is from Lemma B.5, (14) is from Lemma B.3, (15) is from Lemma B.4, and (16) is from Assumption 4.3. □

Lemma B.7. *Under Assumption 4.1, for any function $\varphi(\boldsymbol{\theta}_t^{s+1})$ which is deterministic for a fixed $\boldsymbol{\theta}_t^{s+1}$:*

$$
\left|\mathbb{E}_{t|s}\left[\left\langle \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]-\mathbb{E}_{t|s}\left[\left\langle \nabla J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]\right|\leq\frac{1}{2N}\mathbb{V}ar_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{1}{2}\mathbb{E}_{t-1|s}\left[\left\|\varphi(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]
$$

Proof.

$$
\begin{aligned}
\mathbb{E}_{t|s}\left[\left\langle \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]&=\mathbb{E}_{t|s}\left[\left\langle \mathbb{W}J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]+\mathbb{E}_{t-1|s}\left[\left\langle e_s,\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right] \qquad (17)\\
&=\mathbb{E}_{t|s}\left[\left\langle \nabla J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]+\mathbb{E}_{t-1|s}\left[\left\langle e_s,\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right] \qquad (18)\\
&=\mathbb{E}_{t|s}\left[\left\langle \nabla J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]\\
&\quad+\left\langle \mathbb{E}_{t|s}\left[e_s\right],\mathbb{E}_{t|s}\left[\varphi(\boldsymbol{\theta}_t^{s+1})\right]\right\rangle+\mathbb{C}\mathrm{ov}_{t-1|s}\left(\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s),\varphi(\boldsymbol{\theta}_t^{s+1})\right) \qquad (19)\\
&=\mathbb{E}_{t|s}\left[\left\langle \nabla J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]\\
&\quad+\mathbb{C}\mathrm{ov}_{t-1|s}\left(\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s),\varphi(\boldsymbol{\theta}_t^{s+1})\right) \qquad (20)
\end{aligned}
$$

where (17) is from Definition B.5; (18) is from the fact that $\mathbb{W}J(\boldsymbol{\theta}_t^{s+1})$ is both unbiased and independent from $\varphi(\boldsymbol{\theta}_t^{s+1})$ w.r.t. the sampling at time t alone, which is not true for $\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})$; (19) is from the fact that $\nabla J(\widetilde{\boldsymbol{\theta}}^s)$ is constant w.r.t. $\mathbb{V}\mathrm{ar}_s\left[\cdot\right]$; (20) is from $\mathbb{E}_{t|s}\left[e_s\right]=0$. Hence:

$$
\begin{aligned}
\left|\mathbb{E}_{t|s}\left[\left\langle \blacktriangledown J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]-\mathbb{E}_{t|s}\left[\left\langle \nabla J(\boldsymbol{\theta}_t^{s+1}),\varphi(\boldsymbol{\theta}_t^{s+1})\right\rangle\right]\right|&=\left|\mathbb{C}\mathrm{ov}_{t-1|s}\left(\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s),\varphi(\boldsymbol{\theta}_t^{s+1})\right)\right|\\
&\leq\sqrt{\mathbb{V}\mathrm{ar}_s\left[\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)\right]}\cdot\sqrt{\mathbb{V}\mathrm{ar}_{t-1|s}\left[\varphi(\boldsymbol{\theta}_t^{s+1})\right]} \qquad (21)\\
&\leq\frac{1}{2}\mathbb{V}\mathrm{ar}_s\left[\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{1}{2}\mathbb{V}\mathrm{ar}_{t-1|s}\left[\varphi(\boldsymbol{\theta}_t^{s+1})\right] \qquad (22)\\
&=\frac{1}{2N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{1}{2}\mathbb{V}\mathrm{ar}_{t-1|s}\left[\varphi(\boldsymbol{\theta}_t^{s+1})\right] \qquad (23)\\
&\leq\frac{1}{2N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\frac{1}{2}\mathbb{E}_{t-1|s}\left[\left\|\varphi(\boldsymbol{\theta}_t^{s+1})\right\|^2\right], \qquad (24)
\end{aligned}
$$

where (21) comes from Cauchy-Schwarz inequality

$$
\left|\,\mathbb{C}\mathrm{ov}(X,Y)\right|=\left|\,\mathbb{E}\left[(X-\mu_X)(Y-\mu_Y)^{\mathrm{T}}\right]\right|\leq\mathbb{E}\left[(X-\mu_X)^2\right]^{1/2}\mathbb{E}\left[(Y-\mu_Y)^2\right]^{1/2}=\sqrt{\mathbb{V}\mathrm{ar}(X)\,\mathbb{V}\mathrm{ar}(Y)},
$$

(22) is from Young's inequality, (23) is from the definition of $\widehat{\nabla}_N J(\boldsymbol{\theta})$, and (24) is from Lemma B.5. □

Lemma B.8. *Under Assumptions 4.1 ans 4.3, the expected squared norm of the true gradient $\nabla J(\boldsymbol{\theta}_t^{s+1})$, for appropriate choices of $\alpha_t \geq 0$ and $\beta_t > 0$, can be bounded as follows:*

$$\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] \leq \frac{R_{t+1}^{s+1} - R_t^{s+1}}{\Psi_t} + \frac{d_t V}{N\Psi_t} + \frac{f_t W}{B\Psi_t},$$

where

$$\begin{aligned}
R_t^{s+1} &:= \mathbb{E}_{t-1|s}\left[J(\boldsymbol{\theta}_t^{s+1}) - c_t\left\|\boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2\right],\\
c_m &= 0,\\
c_t &= c_{t+1}\left(1 + \alpha_t\beta_t + \alpha_t + \frac{\alpha_t^2 L^2}{B}\right) + \frac{\alpha_t^2 L^3}{2B},\\
\Psi_t &= \alpha_t\left(\frac{1}{2} - \frac{c_{t+1}}{\beta_t} - \frac{\alpha_t L}{2} - \alpha_t c_{t+1}\right),\\
d_t &= \frac{\alpha_t}{2}\left(1 + 2c_{t+1} + \alpha_t L + 2\alpha_t c_{t+1}\right),\\
f_t &= \alpha_t^2\frac{\Gamma(L + 2c_{t+1})}{2},
\end{aligned}$$

where $L = \max\{L_J, L_g\}$, i.e., the greater of the Lipschitz constants from Lemmas B.2 and B.3.

In particular, the following constraints on α_t and β_t are sufficient:

$$\begin{aligned}
0 \leq \alpha_t &< \frac{1 - {}^{2c_{t+1}}/_{\beta_t}}{L + 2c_{t+1}}\\
\beta_t &> 2c_{t+1}.
\end{aligned}$$

Proof. We have:

$$\begin{aligned}
\mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_{t+1}^{s+1})\right] &\geq \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_t^{s+1}) + \left\langle\nabla J(\boldsymbol{\theta}_t^{s+1}), \boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\right\rangle - \frac{L}{2}\left\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\right\|^2\right] &(25)\\
&= \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_t^{s+1}) + \alpha_t\left\langle\nabla J(\boldsymbol{\theta}_t^{s+1}), \blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\rangle - \frac{\alpha_t^2 L}{2}\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] &(26)\\
&\geq \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_t^{s+1}) + \alpha_t\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2 - \frac{\alpha_t^2 L}{2}\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]\\
&\quad - \frac{\alpha_t}{2N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right] - \frac{\alpha_t}{2}\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right], &(27)
\end{aligned}$$

where (25) is from the L-smoothness of $J(\boldsymbol{\theta})$ (Nesterov, 2013) and (26) is from the SVRPG update. Inequality 27 follows from Lemma B.7 by noticing that $\nabla J(\boldsymbol{\theta}_t^{s+1})$ is a deterministic function given $\boldsymbol{\theta}_t^{s+1}$. As a consequence, we can directly apply Lemma B.7 with $\boldsymbol{\phi}(\boldsymbol{\theta}_t^{s+1}) := \nabla J(\boldsymbol{\theta}_t^{s+1})$.

Next, we have:

$$\begin{aligned}
\mathbb{E}_{t|s}\left[\left\|\boldsymbol{\theta}_{t+1}^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2\right] &= \mathbb{E}_{t|s}\left[\left\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1} + \boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2\right]\\
&= \mathbb{E}_{t|s}\left[\left\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\right\|^2 + \left\|\boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2 + 2\left\langle\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}, \boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\rangle\right] &(28)\\
&= \mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2 + \left\|\boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2 + 2\alpha_t\left\langle\blacktriangledown J(\boldsymbol{\theta}_t^{s+1}), \boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\rangle\right] &(29)\\
&\leq \mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2 + \left\|\boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\|^2 + 2\alpha_t\left\langle\nabla J(\boldsymbol{\theta}_t^{s+1}), \boldsymbol{\theta}_t^{s+1} - \widetilde{\boldsymbol{\theta}}^s\right\rangle\right]
\end{aligned}$$

$$+\frac{\alpha_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right] \tag{30}$$

$$\leq \mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+2\alpha_t\mathbb{E}_{t-1|s}\left[\left|\left\langle\nabla J(\boldsymbol{\theta}_t^{s+1}),\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\rangle\right|\right]$$
$$+\frac{\alpha_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]$$

$$\leq \mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+2\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|\right]$$
$$+\frac{\alpha_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]$$

$$\leq \mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right]+2\alpha_t\mathbb{E}_{t-1|s}\left[\frac{1}{2\beta_t}\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\frac{\beta_t}{2}\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right] \tag{31}$$
$$+\frac{\alpha_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\widetilde{\boldsymbol{\theta}}^s)\right]+\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right], \tag{32}$$

where (28) is obtained using the triangular inequality, (29) is from the SVRPG update, (30) is from Lemma B.7 with $\boldsymbol{\phi}(\boldsymbol{\theta}_t^{s+1}) \coloneqq \boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}^s$, (31) is from Cauchy-Schwarz inequality, and (32) is from Young's inequality in the 'Peter-Paul' variant. Let us consider the following function:

$$R_{t+1}^{s+1} \coloneqq \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_{t+1}^{s+1})-c_{t+1}\left\|\boldsymbol{\theta}_{t+1}^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right]. \tag{33}$$

The objective is now to provide a lower bound to it.

$$R_{t+1}^{s+1} \geq \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_t^{s+1})-\frac{\alpha_t^2 L}{2}\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\mathop{\mathbb{E}}_{t-1|s}\left[\frac{\alpha_t}{2}\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]$$
$$-\frac{\alpha_t}{2N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right]-c_{t+1}\mathbb{E}_{t|s}\left[\left\|\boldsymbol{\theta}_{t+1}^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right] \tag{34}$$

$$\geq \mathbb{E}_{t|s}\left[J(\boldsymbol{\theta}_t^{s+1})-\frac{\alpha_t^2 L}{2}\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\mathbb{E}_{t-1|s}\left[\frac{\alpha_t}{2}\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]-\frac{\alpha_t}{2N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right]$$
$$-c_{t+1}\mathbb{E}_{t|s}\left[\alpha_t^2\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right]$$
$$-2c_{t+1}\alpha_t\mathbb{E}_{t-1|s}\left[\frac{1}{2\beta_t}\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2+\frac{\beta_t}{2}\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right]$$
$$-c_{t+1}\frac{\alpha_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right]-c_{t+1}\alpha_t\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right] \tag{35}$$

$$=\mathbb{E}_{t-1|s}\left[J(\boldsymbol{\theta}_t^{s+1})\right]-c_{t+1}\left(1+\alpha_t\beta_t+\alpha_t\right)\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}\right\|^2\right]$$
$$-\alpha_t^2\left(\frac{L}{2}+c_{t+1}\right)\mathbb{E}_{t|s}\left[\left\|\blacktriangledown J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{\alpha_t}{2}\left(1-\frac{2c_{t+1}}{\beta_t}\right)\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]$$
$$-\frac{\alpha_t}{2N}\left(1+2c_{t+1}\right)\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right]$$

$$\geq \mathbb{E}_{t-1|s}\left[J(\boldsymbol{\theta}_t^{s+1})\right]-c_{t+1}\left(1+\alpha_t\beta_t+\alpha_t\right)\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}\right\|^2\right]$$
$$-\alpha_t^2\left(\frac{L}{2}+c_{t+1}\right)\left(\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]+\frac{1}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right]\right.$$
$$\left.+\frac{L^2}{B}\mathbb{E}_{t-1|s}\left[\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}^s\right\|^2\right]+\frac{\Gamma W}{B}\right)+\frac{\alpha_t}{2}\left(1-2\frac{c_{t+1}}{\beta_t}\right)\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]$$
$$-\frac{\alpha_t}{2N}\left(1+2c_{t+1}\right)\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right] \tag{36}$$

$$
\begin{aligned}
&= \mathbb{E}_{t-1|s}\left[J(\boldsymbol{\theta}_t^{s+1}) - \left(c_{t+1}\left(1+\alpha_t\beta_t+\alpha_t+\frac{\alpha_t^2L^2}{B}\right)+\frac{\alpha_t^2L^3}{2B}\right)\left\|\boldsymbol{\theta}_t^{s+1}-\tilde{\boldsymbol{\theta}}\right\|^2\right] \\
&\quad + \alpha_t\left(\frac{1}{2}-\frac{c_{t+1}}{\beta_t}-\frac{\alpha_t L}{2}-\alpha_t c_{t+1}\right)\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] \\
&\quad - \frac{\alpha_t}{2N}\left(1+2c_{t+1}+\alpha_t L+2\alpha_t c_{t+1}\right)\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right] - \alpha_t^2\frac{(L+2c_{t+1})\Gamma W}{2B} \\
&= R_t^{s+1} + \Psi_t\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] - \frac{d_t}{N}\mathbb{V}\mathrm{ar}_s\left[g(\cdot|\tilde{\boldsymbol{\theta}}^s)\right] - \frac{f_t}{B}W, \\
&\geq R_t^{s+1} + \Psi_t\mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] - \frac{d_t}{N}V - \frac{f_t}{B}W, \qquad (37)
\end{aligned}
$$

where (34) is from (27) noticing that $\mathbb{E}_{t|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right] = \mathbb{E}_{t-1|s}\left[\left\|\nabla J(\boldsymbol{\theta}_t^{s+1})\right\|^2\right]$, (35) is from (32), (36) is from Lemma B.6, and (37) is from Assumption 4.2. To complete the proof, besides rearranging terms, we have to ensure that $\Psi_t > 0$ for each t. This gives the constraints on α_t and β_t. □

Main theorem

We finally provide the proof of the convergence theorem:

Theorem 4.4 (Convergence of the SVRPG algorithm)**.** *Assume the REINFORCE or the G(PO)MDP gradient estimator is used in SVRPG (see Equation (4)). Under Assumptions 4.1, 4.2 and 4.3, the parameter vector $\boldsymbol{\theta}_A$ returned by Algorithm 2 after $T = m \times S$ iterations has, for some positive constants ψ, ζ, ξ and for proper choice of the step size α and the epoch size m, the following property:*

$$
\mathbb{E}\left[\|\nabla J(\boldsymbol{\theta}_A)\|_2^2\right] \leq \frac{J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}_0)}{\psi T} + \frac{\zeta}{N} + \frac{\xi}{B},
$$

where $\boldsymbol{\theta}^*$ is a global optimum and ψ, ζ, ξ depend only on G, F, V, W, α and m.

Proof. We prove the theorem for the following values of the constants:

$$
\psi \coloneqq \min_t\{\Psi_t\}, \qquad \zeta \coloneqq \frac{\max_t\{d_t\}V}{\psi}, \qquad \xi \coloneqq \frac{\max_t\{f_t\}W}{\psi},
$$

where Ψ, d_t and f_t are defined in Lemma B.8. Starting from Lemma B.8, summing over iterations of an epoch s and using telescopic sum we obtain

$$
\begin{aligned}
\sum_{t=0}^{m-1}\mathbb{E}_{t|s}\left[\left\|\nabla J((\boldsymbol{\theta}_t^{s+1}))^2\right\|\right] &\leq \frac{\sum_{t=0}^{m-1}\left(R_{t+1}^{s+1}-R_t^{s+1}\right)}{\psi} + \frac{m\zeta}{N} + \frac{m\xi}{B} \\
&= \frac{R_m^{s+1}-R_0^{s+1}}{\psi} + \frac{m\zeta}{N} + \frac{m\xi}{B}
\end{aligned}
$$

By using the definition of R_t^s in (33), the fact that $c_m = 0$ and $\boldsymbol{\theta}_0^{s+1} = \widetilde{\boldsymbol{\theta}}^s = \boldsymbol{\theta}_m^s$, we can state that:

$$
\begin{aligned}
R_m^{s+1} - R_0^{s+1} &= \mathop{\mathbb{E}}_{m|s}\left[J(\boldsymbol{\theta}_m^{s+1}) - c_m\left\|\boldsymbol{\theta}_m^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right] - \mathop{\mathbb{E}}_{0|s}\left[J(\boldsymbol{\theta}_0^{s+1}) - c_0\left\|\boldsymbol{\theta}_0^{s+1}-\widetilde{\boldsymbol{\theta}}^s\right\|^2\right] \\
&= \mathop{\mathbb{E}}_{m|s}\left[J(\boldsymbol{\theta}_m^{s+1})\right] - \mathop{\mathbb{E}}_{0|s}\left[J(\widetilde{\boldsymbol{\theta}}^s)\right] = \mathop{\mathbb{E}}_{m|s}\left[J(\widetilde{\boldsymbol{\theta}}^{s+1}) - J(\widetilde{\boldsymbol{\theta}}^s)\right]
\end{aligned}
$$

Next, summing over epochs:

$$\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}_{t|s}\left[\left\|\nabla J((\boldsymbol{\theta}_t^{s+1}))^2\right\|\right] \leq \frac{\sum_{s=0}^{S-1}\mathbb{E}_{m|s}\left[J(\tilde{\boldsymbol{\theta}}^{s+1}) - J(\tilde{\boldsymbol{\theta}}^{s})\right]}{\psi} + \frac{T\zeta}{N} + \frac{T\xi}{B}$$

$$\leq \frac{\mathbb{E}\left[J(\tilde{\boldsymbol{\theta}}^{S}) - J(\tilde{\boldsymbol{\theta}}^{0})\right]}{\psi} + \frac{T\zeta}{N} + \frac{T\xi}{B} \tag{38}$$

$$\leq \frac{J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}^0)}{\psi} + \frac{T\zeta}{N} + \frac{T\xi}{B}, \tag{39}$$

where the expectation in (38) is w.r.t. all the trajectories sampled in a run of Algorithm 2 and (39) is from the definition of $\boldsymbol{\theta}^*$ (i.e., the policy performance maximizer). Finally, we consider the expectation w.r.t. all sources of randomness, including the uniform sampling of the output parameter:

$$\mathbb{E}\left[\left\|\nabla J((\boldsymbol{\theta}_t^{s+1}))\right\|^2\right] = \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}_{t|s}\left[\left\|\nabla J((\boldsymbol{\theta}_t^{s+1}))\right\|^2\right] \leq \frac{J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}^0)}{\psi T} + \frac{\zeta}{N} + \frac{\xi}{B}.$$

□

C. Applicability to Gaussian Policies

We provide more details on the applicability of Theorem 4.4 on the case of Gaussian policies. We start from the case of one-dimensional bounded action space $\mathcal{A} \subset \mathbb{R}$, linear mean $\mu(s) = \boldsymbol{\theta}^T\boldsymbol{\phi}(s)$ and fixed standard deviation σ:

$$\pi_{\boldsymbol{\theta}}(a|s) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{(\boldsymbol{\theta}^T\boldsymbol{\phi}(s) - a)^2}{2\sigma^2}\right\},$$

where $\boldsymbol{\phi}(s) \leq M_\phi$ is a bounded feature vector, and we see under which conditions the three assumptions of Section 4 hold.

Assumption 4.1 (On policy derivatives)**.** *For each state-action pair (s, a), any value of $\boldsymbol{\theta}$, and all parameter components i, j there exist constants $0 \leq G, F < \infty$ such that:*

$$|\nabla_{\theta_i}\log\pi_{\boldsymbol{\theta}}(a|s)| \leq G, \qquad \left|\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log\pi_{\boldsymbol{\theta}}(a|s)\right| \leq F.$$

For the Gaussian policy defined above, it's easy to show that:

$$\nabla_{\theta_i}\log\pi_{\boldsymbol{\theta}}(\tau) = \phi_i(s)\frac{a - \boldsymbol{\theta}^T\phi(s)}{\sigma^2},$$
$$\frac{\partial^2}{\partial\theta_i\partial\theta_j}\log\pi_{\boldsymbol{\theta}}(\tau) = \frac{\phi_i(s)\phi_j(s)}{\sigma^2}.$$

Hence, Assumption 4.1 is automatically satisfied [10] by taking $G = \frac{M_\phi|\mathcal{A}|}{\sigma^2}$ and $F = \frac{M_\phi^2}{\sigma^2}$.

Assumption 4.2 (On the variance of the gradient estimator)**.** *There is a constant $V < \infty$ such that, for any policy $\pi_{\boldsymbol{\theta}}$:*

$$\mathbb{V}\mathrm{ar}\left[g(\cdot|\boldsymbol{\theta})\right] \leq V.$$

As mentioned, (Pirotta et al., 2013) provides a bound on the variance of the REINFORCE estimator, adapted from (Zhao et al., 2011), which does not depend on $\boldsymbol{\theta}$:

$$\mathbb{V}\mathrm{ar}\left[\widehat{\nabla}_N J(\theta_i)\right] \leq \frac{R^2M_\phi^2H(1-\gamma^H)^2}{N\sigma^2(1-\gamma)^2}.$$

The same authors provide a similar bound for G(PO)MDP.

[10] This relies on the fact that $\boldsymbol{\theta}^T\phi(s)$ lies in bounded $\mathcal{A}$. In practice, this is usually enforced by clipping the action selected by $\pi_{\boldsymbol{\theta}}$. A more rigorous way would be to employ the truncated Gaussian distribution.

Assumption 4.3 (On the variance of importance weights)**.** *There is a constant $W < \infty$ such that, for each pair of policies encountered in Algorithm 2 and for each trajectory,*

$$\mathbb{V}ar\left[\omega(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)\right] \leq W, \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d, \tau \sim p(\cdot|\boldsymbol{\theta}_1).$$

It is noted in (Cortes et al., 2010) that, for any two Gaussian distributions $\mathcal{N}(\mu_1, \sigma_1)$ and $\mathcal{N}(\mu_2, \sigma_2)$, the variance of the importance weights from the latter to the former is bounded whenever $\sigma_2 > \frac{\sqrt{2}}{2}\sigma_1$. This is automatically satisfied by our fixed-variance Gaussian policies, since $\sigma_2 = \sigma_1 = \sigma$.

We now briefly examine some generalizations of the simple Gaussian policy defined above that can be found in applications:

Multi-dimensional actions. When actions are multi-dimensional, factored Gaussian policies are typically employed, so the results extend trivially from the one-dimensional case. Actual multi-variate Gaussian distributions would require more calculations, but we do not expect substantially different results.

Non-linear mean. In complex continuous tasks, $\mu(s)$ often represents a deep neural network, or multi-layer perceptron, where $\boldsymbol{\theta}$ are the weights of the network. The analysis of first and second order log-derivatives in such a scenario is beyond the scope of this paper.

Adaptive variance. It is a common practice to learn also the variance of the policy in order to adapt the degree of exploration. The variance (or diagonal covariance matrix in the multi-dimensional case) can be learned as a separate parameter or be state-dependent like the mean. In any case, adaptive variance must be carefully employed since it can clearly undermine all the three assumptions of Theorem 4.4.

D. Practical SVRPG Versions

We provide more details on the practical variants of SVRPG.

D.1. Adpative Step Size

Algorithm 3 Adam

Input: A gradient estimate g_t and parameters β_1, β_2, ϵ and α.
$\kappa_t = \beta_1 \kappa_{t-1} + (1-\beta_1) g_t$
$\nu_t = \beta_2 \nu_{t-1} + (1-\beta_2) g_t \circ g_t$ ($\circ$ is the Hadamard (component-wise) product)
$\hat{\kappa}_t = \dfrac{\kappa_t}{1-\beta_1^t}$
$\hat{\nu}_t = \dfrac{\nu_t}{1-\beta_2^t}$
$\Delta(g_t) = \dfrac{\alpha}{\sqrt{\hat{\nu}_t} + \epsilon} \hat{\kappa}_t$
Return: The increment $\Delta(g_t)$ of the parameters.

Let us give a deeper insight on the two different learning rate schedules used by our algorithm. We report pseudo-code of the original ADAM (Kingma & Ba, 2014) in Algorithm 3. As mentioned, we use two distinct instances of ADAM to manage different sources of variance: one related to the snapshots, and one to the sub-iterations. In this way the ADAM associated to the snapshots takes into account only the history of gradient moments at the snapshots. By using Algorithm 3 as a subroutine $\mathbf{ADAM}(g, \alpha, \beta)$, we can explicitly define our gradient updates:

$$\boldsymbol{\theta}_1^{s+1} = \widetilde{\boldsymbol{\theta}}^s + \mathbf{ADAM}\left(\widehat{\nabla}_N J(\widetilde{\boldsymbol{\theta}}^s), \beta_1, \beta_2, \alpha\right),$$
$$\boldsymbol{\theta}_{t+1}^{s+1} = \mathbf{ADAM}\Big(\blacktriangledown J(\boldsymbol{\theta}_t^{s+1}), \beta_1, \beta_2, \frac{\alpha}{2}\Big) \text{ for } t = 1, \dots, m-1,$$

where separate histories are kept for estimated first moments κ_{FG}, κ_{IS} and estimated second moments ν_{FG}, ν_{IS}. The meta-parameters α, β_1, β_2 are constant and set to default values or with minor manual tuning (see table 1). Note that we

double the sub-iterations' learning rate for the snapshot ADAM since we can rely on a larger number of trajectories (N instead of B) to control the variance.

D.2. Baseline

The baseline used in the Half Cheetah experiment is the one used in (Duan et al., 2016). It is a linear state-value function estimator, or critic. The (time-varying) feature encoding for the linear baseline is:

$$\boldsymbol{\phi}(s,t) = [s, s \odot s, 0.01t, (0.01t)^2, (0.01t)^3, 1],$$

where $s \in \mathbb{R}^d$ is the state vector and $\odot$ is the element-wise product. The baseline is then:

$$b(s_t, a_t) = \lambda^T \boldsymbol{\phi}(s_t, t).$$

The baseline is fitted from scratch at each policy gradient iteration, with least squares, to match state-value function $V^\pi(s)$. When used with SVRPG, the critic parameter λ is updated only at the snapshot.

E. Experimental Details

We describe the RL tasks of Section 7 in more detail:

1. *Cart-Pole Balancing* : an inverted pendulum mounted on a cart must be kept standing by moving the cart backward or forward ;4-dimensional state space: cart position x, pole angle θ, cart velocity $\dot{x}$ and pole velocity $\dot{\theta}$; 1-dimensional action space: the horizontal force applied to the cart body. Reward function is defined as $r(s,a) := 10 - (1 - cos(\theta)) - 10^{-5} \|a\|^2$. The episodes terminate when $|x| > 2.4$ or $|\theta| > 0.2$ or the number of time steps T is greater than 100.
2. *Mujoco Swimmer*: a snake-like robot immersed in a fluid must move forward; 13-dimensional state space: 3 links velocities (v_x and v_y of center of masses) and 2 actuated joints angles. 2-dimensional action space: the two momentums applied on actuated joints. The reward function is defined as $r(s,a) := v_x - 10^{-4} \|a\|_2^2$. The episodes terminate when the number of time steps T is greater than 500.
3. *Mujoco Half Cheetah*: a planar biped robot must move forward; 20-dimensional state space: 9 links and 6 actuated joints angles. 6-dimensional action space: the 6 momentums applied on actuated joints. The reward function is defined as $r(s,a) := v_x - 0.05 \|a\|_2^2$. The episodes terminate when the number of time steps T is greater than 500.

All the parameters used in the experiments, including neural network architectures, are reported in the following table:

Table 1: Parameters used in the experiments of Section 7.

	Cart-Pole	Swimmer	Half Cheetah
NN hidden weights	8	32x32	100x50x25
NN activation	tanh	tanh	tanh
Adam α (SVRPG)	$5 \cdot 10^{-2}$	10^{-3}	10^{-3}
Adam α (GPOMDP)	10^{-2}	10^{-3}	10^{-2}
Adam β_1	0.9	0.9	0.9
Adam β_2	0.99	0.99	0.99
Snapshot batch size N (SVRPG)	100	100	100
Mini-batch size B (SVRPG)	10	10	10
Batch size (GPOMDP)	10	10	100
Max number of sub-iterations	50	20	20
Task horizon	100	500	500
Baseline	No	No	Yes
Discount factor γ	0.99	0.995	0.99
Total number of trajectories	10000	20000	50000

Where not specified, meta-parameters are shared among G(PO)MDP and SVRPG. Refer to (Duan et al., 2016) for more details about G(PO)MDP (REINFORCE in the paper) on the Half Cheetah task.

Videos: The supplementary materials include videos showing the behavior of the final SVRPG agents on the three considered tasks.